

ROBOT MOBILITY - Then and Now

SERVO

FOR THE ROBOT INNOVATOR

www.servomagazine.com

MAGAZINE

April 2010

2 Part Series

SO YOU WANT TO BUILD A COMBOT

*Basics of designing
a featherweight
Combat Robot from
the ground up.*



U.S. \$5.50 CANADA \$7.00



0 71486 02422 4

♦ **Trade two PWM
channels on your transmitter
for a whole bunch of switches.**

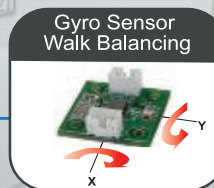
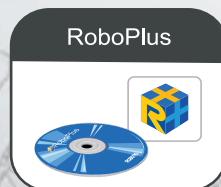
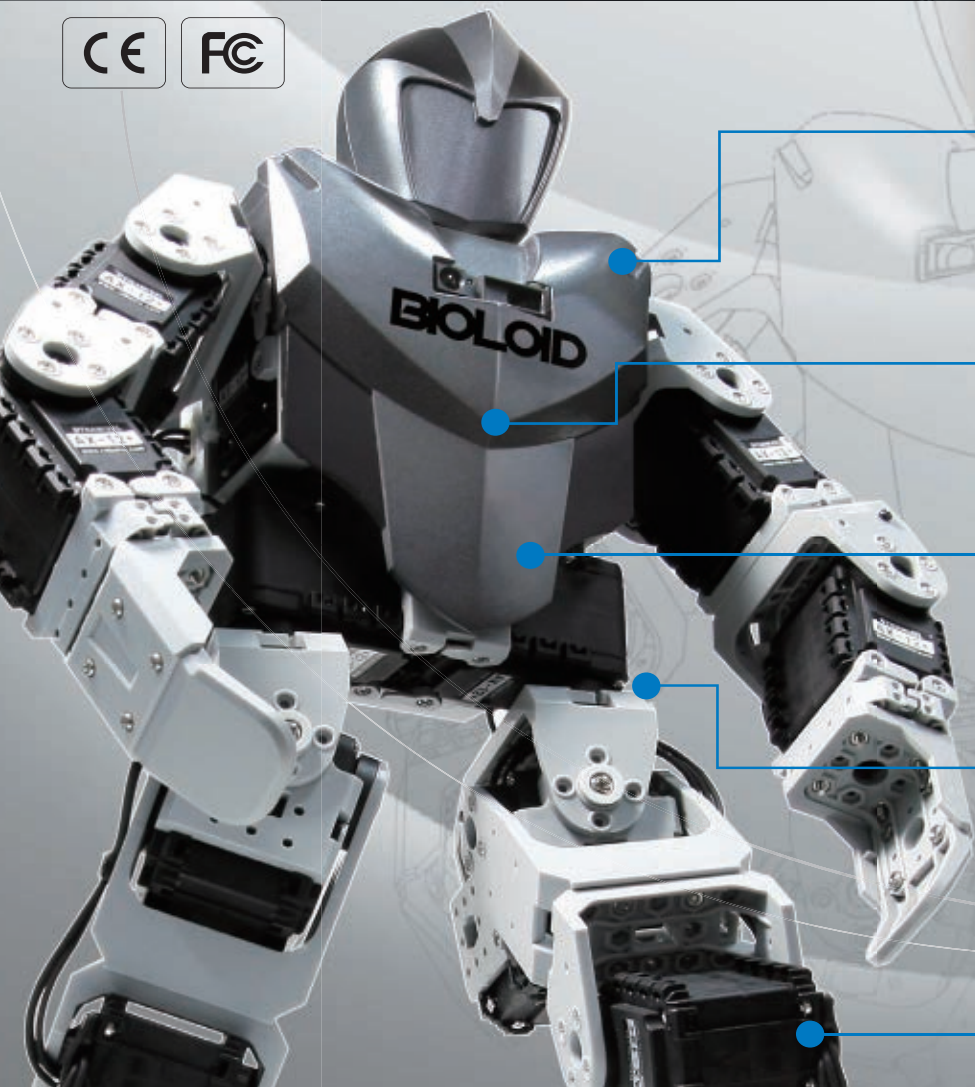
The Evolution Continues

BIOLOID

PREMIUM Kit

Authorized kit by **ROBO ONE**

Possible to build diverse type of robots and program by yourself
Optimized material for robot education class



DYNAMIXEL

High-performance robot exclusive actuator

ROBOTIS
www.robotis.com

NEW AX-12+ / AX-18F



	AX-12+	AX-18F
Weight(g) / (oz)	53.5(g) / 1.88(oz)	54.5(g) / 1.92(oz)
Dimension(mm) / (inch)	32×50.1×40(mm) 1.25×1.97×1.57(inch)	32×50.1×40(mm) 1.25×1.97×1.57(inch)
Gear Ratio(material)	1 : 254(enpla)	1 : 254(enpla)
Operation Voltage(V)	9~12	9~12
Holding Torque(kgf·cm)	15 at 12V / 1.5A	18 at 12V / 2.2A
No load speed(RPM)	59	97
Network Interface	TTL	TTL
Position Sensor(Resolution)	Potentiometer(300°/1024)	Potentiometer(300°/1024)
Motor	Cored Motor	Coreless Motor

NEW RX-64



	RX-64
Weight(g) / (oz)	125(g) / 4.4(oz)
Dimension(mm) / (inch)	40.1×61.3×45.8(mm) 1.57×2.41×1.8(inch)
Gear Ratio(material)	1 : 200(metal)
Operation Voltage(V)	12~18.5
Holding Torque(kgf·cm)	52 at 18.5V / 2.6A
No load speed(RPM)	64
Network Interface	RS-485
Position Sensor(Resolution)	Potentiometer(300°/1024)
Motor	Maxon Motor

NEW RX-24F/ RX-28



	RX-24F	RX-28
Weight(g) / (oz)	67(g) / 2.36(oz)	72(g) / 2.53(oz)
Dimension(mm) / (inch)	35.5×50.8×41.8(mm) 1.39×2×1.64(inch)	35.5×50.8×41.8(mm) 1.39×2×1.64(inch)
Gear Ratio(material)	1 : 193(metal)	1 : 193(metal)
Operation Voltage(V)	9~12	12~18.5
Holding Torque(kgf·cm)	26 at 12V / 2.4A	37 at 18.5V / 1.9A
No load speed(RPM)	126	67
Network Interface	RS-485	RS-485
Position Sensor(Resolution)	Potentiometer(300°/1024)	Potentiometer(300°/1024)
Motor	Coreless Motor	Maxon Motor

NEW EX-106+



	EX-106+
Weight(g) / (oz)	154(g) / 5.43(oz)
Dimension(mm) / (inch)	40.1×65.3×50.1(mm) 1.57×2.57×1.97(inch)
Gear Ratio(material)	1 : 184(metal)
Operation Voltage(V)	12~18.5
Holding Torque(kgf·cm)	107 at 18.5V / 7A
No load speed(RPM)	91
Network Interface	RS-485
Position Sensor(Resolution)	Magnetic encoder(251°/4096)
Motor	Maxon Motor

The Future of Servo Control is Calling...

ServoCenter™

Features

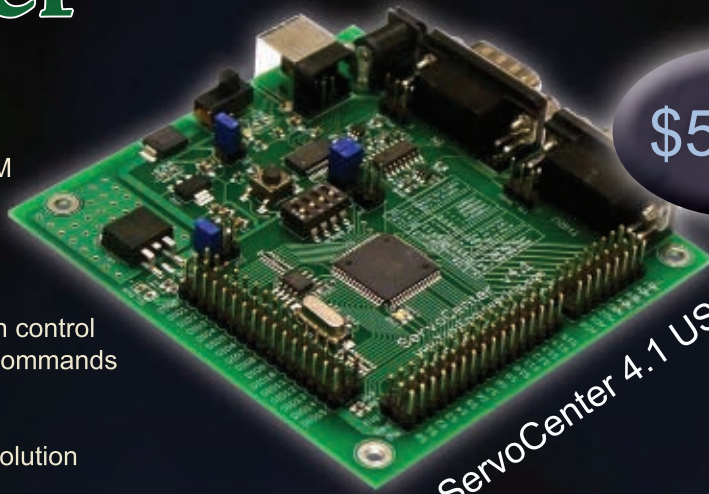
- USB, RS232, and TTL serial support
- 16 servos, 16 digital I/O, 8 analog inputs
- Built-in SC-BASIC Sequencer with EEPROM
- Sequencer allows stand-alone operation
- 64 scene presets stored in EEPROM
- Presets instantly loaded or cross-faded
- Built-in configurable smoothing algorithm
- Independent, simultaneous speed & position control
- Scaled, percentage, and group movement commands
- Timed movement commands
- Max, min, & startup position settings
- Ultra-precise 0.05425µS jitter-free pulse resolution

Flexibility

- User upgradeable firmware
- Upload your own firmware with bootloader
- Watchdog timer for failsafe operation
- Over-current, over-temperature, polarity protection
- Internal regulator, external power, or battery power options
- Supports 4.8/6.0V regulated servo supply voltages at 5 Amps
- Each digital I/O and analog input has supply pins
- Direct serial, activeX control, or Win32 DLL communication
- Programming examples in 10+ languages
- Windows, Linux, Mac OSX compatible

Free Control Panel Software

- Easy editing and configuration of servo settings
- Configure and set up digital I/O & ADC settings
- Edit scene presets
- Program the SC-BASIC sequencer
- Upload and run your SC-BASIC programs
- Debug & communicate with terminal window



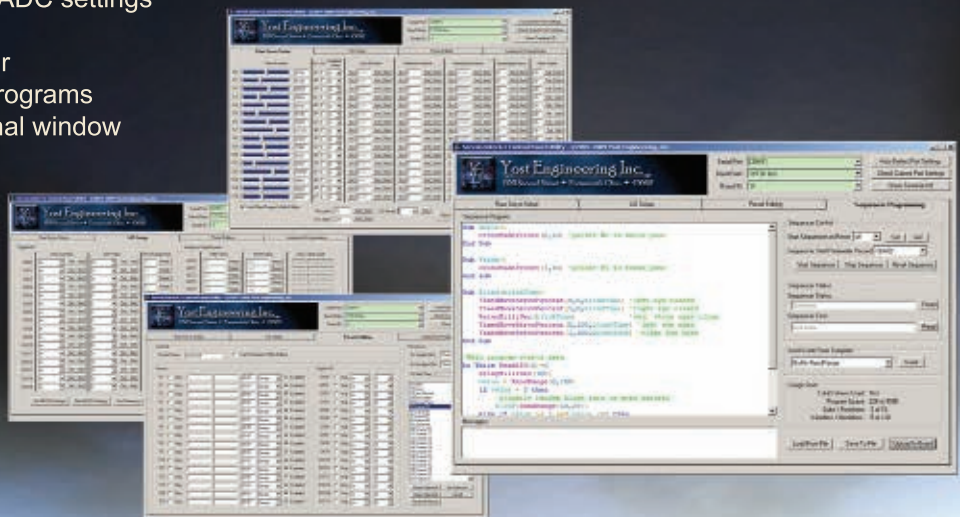
\$56.99

ServoCenter 4.1 USB



\$59.99

ServoCenter 4.1
USB Mini



www.Servo-Center.com

Yost Engineering Inc.

The Combat Zone...

Features

- 28 BUILD REPORT:**
Crossfire — From Broken Robot to Breaking Robots
- 30 MANUFACTURING:**
Keeping Your Stuff Together
- 33 PARTS IS PARTS:**
FingerTech Robotics' Spark Motor Series
- 33 How Not to Die.**
I Mean, Build a Test Box
- 35 Combat Zone's**
Greatest Hits

Events

- 32 Results/Upcoming**
Events



PAGE 10

Columns

- 08 Robytes**
by Jeff Eckert
Stimulating Robot Tidbits
- 10 GeerHead**
by David Geer
Penbo for Girls, Prime-8 for Boys
- 13 Ask Mr. Roboto**
by Dennis Clark
Your Problems Solved Here
- 76 Then and Now**
by Tom Carroll
Robot Mobility



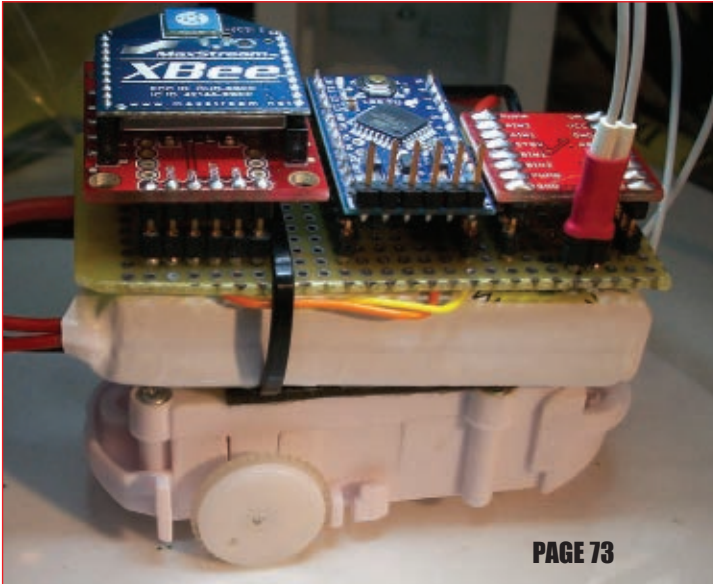
PAGE 35



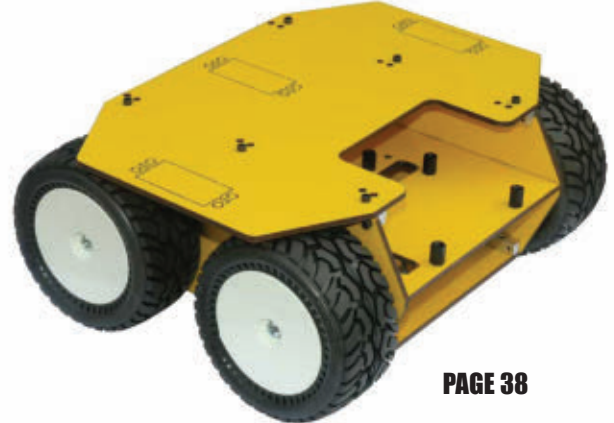
PAGE 76

Departments

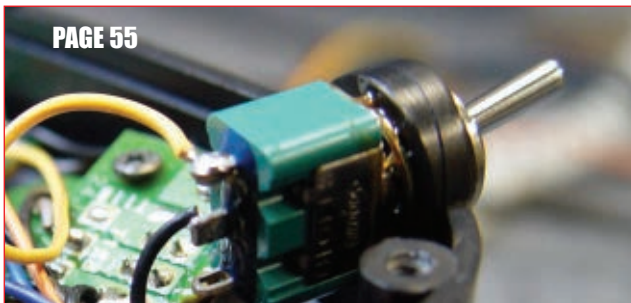
- | | |
|---------------------------|--------------------------|
| 06 Mind/Iron | 27 Showcase |
| 07 Bio-Feedback | 64 SERVO Webstore |
| 20 Events Calendar | 81 Robo-Links |
| 22 New Products | 81 Advertiser's |
| 24 Bots in Brief | Index |



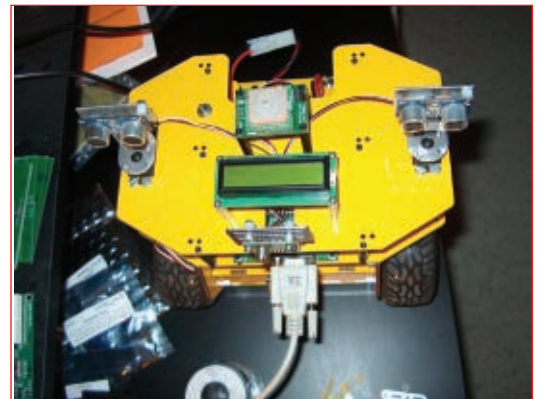
PAGE 73



PAGE 38



PAGE 55



38 GPS Navigation — Part 2

by Chris Savage

Build a small robot to test and prove the theories discussed in Part 1.

42 The Bioloid Premium Kit Wrap-Up

by Rob Farrell

In the Feb. '10 issue, we covered the hardware components and assembly of the Bioloid Premium. This time, we'll take a look at the vast capabilities of the RoboPlus software that comes with the kit.

48 Look Who's Talking!

by Fred Eady

Explore this simple method that lets you and your robot play together in a wireless world.

55 Trade Two PWM Channels for a Whole Bunch of Switches

by Jim Miller

See how to perform surgery on a transmitter to gain more on/off switches and data from your rover.

67 So, You Want to Build a ComBot — Part 1

by Greg Intermaggio

This short series will cover the basics of designing a featherweight ComBot from the ground up using only tools you'd find in a home shop.

73 Swarm Robots and Sensor Virtualization

by Mike Keesling

Ben has nothin' on our Zhu Zhu hamsters when it comes to working together in a pack.

Mind / Iron

by Bryan Bergeron, Editor



Robotics Horizon

I recently attended a review of robotics research in New England's universities which included a tour of seven major robotics labs at MIT. Given that academia is typically a year or two ahead of what's deployable by the military and several years ahead of what may be practical in the marketplace, the future looks promising.

Highlights of the meeting included surgical navigation and surgical robotics — two nascent technologies in medicine. You've probably heard of the Da Vinci surgical robot (www.davincisurgery.com) which is increasingly used in specific elective surgeries. The Da Vinci is a remote-controlled robot laden with sensors, motors, and minute effectors, but without autonomous functions. Benefits of the system include shorter hospital stays — an important factor in containing medical costs. Pending clearance by the Food and Drug Administration (FDA), several other robotic platforms should be coming on line shortly.

Surgical Navigation systems such as the Medtronic StealthStation (www.medtronicnavigation.com) are also gaining in popularity. These systems use optical and RF tracking to enable a surgeon to visualize the position of surgical instruments relative to a patient's anatomy in 3D space. The advantage of surgical navigation systems is that they enable doctors to see the operating field when the instruments and tissues are hidden from the naked eye. Properly employed, these systems can reduce surgical

errors, and enable a surgeon to carry out a procedure with a relatively small incision.


One of the more interesting robotic technologies on the horizon was soft machines, such as those under development at the Tufts Biomimetic Devices Laboratory (ase.tufts.edu/bdl). While most of the robotics world is working with and thinking about traditional hardware sensors and effectors, researchers in this lab are working with elastomers, fluids, artificial muscles, and nanosensors.

Instead of a human or lobster, the biomimetic model for much of the work on soft robots is a soft, low-density, tactile-sensitive caterpillar. The potential advantages of a lightweight, soft robot over a heavy robot with a rigid skeleton or exoskeleton include the ability to climb textured surfaces, crawl along ropes and wires, and burrow into winding, confined spaces. Imagine a few dozen soft robots released into the rubble following a devastating earthquake. Soft rescue bots should have a much easier time snaking toward victims relative to traditional robots constrained by stiff, physical bulk.

Soft-bodied robots have obvious military applications, as well. Over a year ago, the DOD offered a small business innovation research (SBIR) grant for the purpose of developing a soft robot that could push itself through a crack in a barrier and reassemble intact on the other side. The endpoint of this R&D effort was an ordnance delivery system that could squeeze through a crack and deliver its payload on the other side of a wall. I've seen prototypes of soft delivery vehicles, including a multi-chambered sphere controlled by pneumatics. Pressurizing some of the elastic chambers while reducing pressure in others resulted in steerable movement. There was no room for a significant payload, however.

So, what's my take-away from the meeting? I'm certain that one day you'll be able to walk to your corner drugstore and have an autonomous robotic surgeon sew up a laceration, lance a blister, or remove an ingrown toenail. I'm also confident that the military will perfect soft robotic weapons. The underlying challenge in these and other robotics R&D projects is economics. Regardless of the state of the economy, there will always be competition from traditional technologies, the need for a positive return on investment, and the need for visionaries to champion technologies to the forefront. So, step up and let's get going. **SV**

2009 CD ROM



Contents include

- January through December 2009 issues of **SERVO Magazine**
- Supporting code and media files from each issue
- Windows compatible Adobe Reader
- MAC compatible Adobe Reader

On Sale Now!

Only \$24.95

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct., Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore Only **1-800-783-4624**
www.servomagazine.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277, N. Hollywood, CA 91615

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

**ASSOCIATE PUBLISHER/
VP OF SALES/MARKETING**
Robin Lemieux
display@servomagazine.com

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS

Jeff Eckert	Jenn Eckert
Tom Carroll	David Geer
Dennis Clark	R. Steven Rainwater
Fred Eady	Kevin Berry
Gregory Intermaggio	Chris Savage
Rob Farrell	Mike Keesling
Jim Miller	Nick Martin
Matthew Spurk	James Baker

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

**MARKETING COORDINATOR
WEBSTORE**
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Shannon Christensen

Copyright 2010 by
T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. *SERVO Magazine* assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in *SERVO*. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in *SERVO*. Please send all editorial correspondence, UPS, overnight mail, and artwork to: **430 Princeland Court, Corona, CA 92879.**

BIO-FEEDBACK

Dear *SERVO*:

Nothing much, but I noticed that the board layout in the CheapBot proximity detector in the Feb. '09 issue shows seven resistors and the schematic shows six. Thought I would point it out. I mill my own boards and ran into the oops during setup.

I enjoy *SERVO*, and use it for ideas for my four grandkids and myself. I believe it is a great source for ideas to inspire and improve their lives.

Don Perry
Chatham Center, NY

Different GEARS for different years

GEARS offers unique Robotics Programs for Middle, High School, College and beyond!

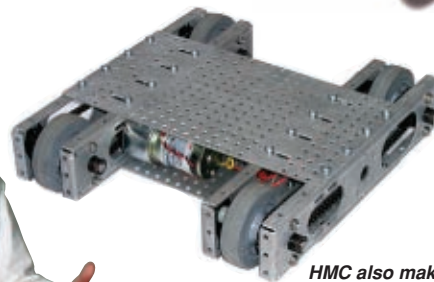
Heavy Metal Chassis

The Heavy Metal Chassis now comes in three versions: HMC-Lite, HMC Belt or Chain Drive, and HMC with articulating wheel base for off-road action. It's a solid foundation for any Robot Program. Students can spend more time integrating their own fabrications, ESC's, microcontrollers and sensors.

Starting at \$399.



HMC-Lite above with optional Machine Science XIPMod Robot Controller and battery. HMC Articulating Robot inset below.



HMC also makes a super combat robot foundation.



HMC 10 lb. chassis supports 200 lbs standing weight!



Robot Starter Kit

Includes robot chassis, servo motors, wheels, and your choice of RC Kit or Micro Controller and Sensor Kit **for just \$229.95.**

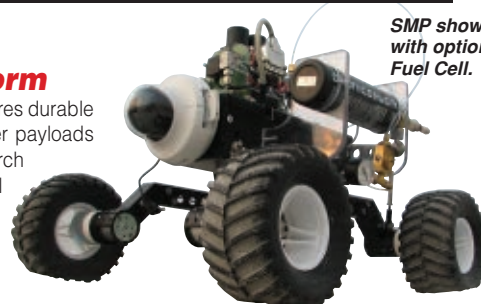
Surface Mobility Platform

The Surface Mobility Platform (SMP) features durable components for larger robots and heavier payloads in off-road environments. It is ideal for search and rescue experimentation, environmental research, surveillance or even simulated first responder activities.

Base kit \$1,500.

Gears

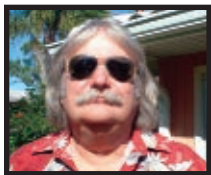
Educational Systems, LLC



SMP shown with optional Fuel Cell.

Contact Mark Newby at **mnewby@gearseds.com**
www.gearseds.com

105 Webster Street, Hanover, MA 02339 • 781.878.1512



New Fleet of AUVs



Underwater view of glider. Photo provided by Holger v. Neuhoff, IFM-GEOMAR.

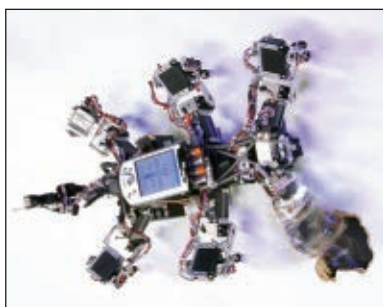
By the time you read this, the Leibniz Institute of Marine Sciences (IFM-GEOMAR; www.ifm-geomar.de) will have launched a new swarm of autonomous underwater vehicles to perform oceanographic

studies in the tropical Atlantic. Although there are already more than 3,000 AUVs drifting around, taking measurements, and relaying gathered information, the present units have at least one major drawback: They have no propulsion systems, and thus their paths are at the mercy of the currents. The new ones are a bit different. Although they have no motors or propellers, they can use their wings to move forward much like sailplanes do in the air. Using a zigzag pattern, one of them can cycle between the surface and depths of about 1,000 m (3,280 ft) using only about as much power as a bicycle headlight. Not only can the new gliders transmit gathered data in real time, researchers can contact them via satellite phones and program them with new mission parameters.

Present plans call for as many as 10 of the gliders simultaneously carrying out autonomous missions for weeks to months, measuring temperature, salinity, oxygen and chlorophyll content, and seawater turbidity. The first two-month mission will take place about 60 mi northeast of the Cape Verde Island of Sao Vicente. If you want to follow their progress, just log onto gliderweb.ifm-geomar.de.

Walkbot Overcomes Obstacles Using "Chaos Control"

As a result of research by scientists at the Georg-August University of Göttingen (www.uni-goettingen.de) with support from the Bernstein Center for Computational Neuroscience and the Max Planck Institute for Dynamics and Self, a six-legged robot has



This robot generally uses regular leg movements but can employ a "chaotic" pattern to free itself. Courtesy of Network Dynamics Group, Max Planck Institute for Dynamics, and Self.

been developed that can autonomously switch among various gaits to move slowly or quickly, overcome obstacles, navigate inclines, free itself from holes, and so on. The secret is a mechanism called "chaos control" that is used to produce the different movement patterns.

Using various sensors, the robot gathers information about its environment and feeds it to the central pattern generator (CPG) which controls its movements. The CPG is a tiny network of simple interconnection elements, comparable with two neural units. The interaction between sensors and the CPG can either be programmed or learned by the robot through experience. The next step will be to equip the little creeper with a motor memory, enabling him to plan his movements in advance.

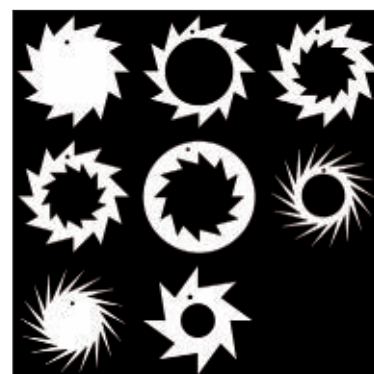
Bacteria Drive for Bots and Other Mechanisms

There is always the question of how you're going to power any kind of self-contained, autonomous device, and a fairly strange answer is under development at Argonne National Lab (www.anl.gov).

Scientists at ANL and Northwestern University have discovered that

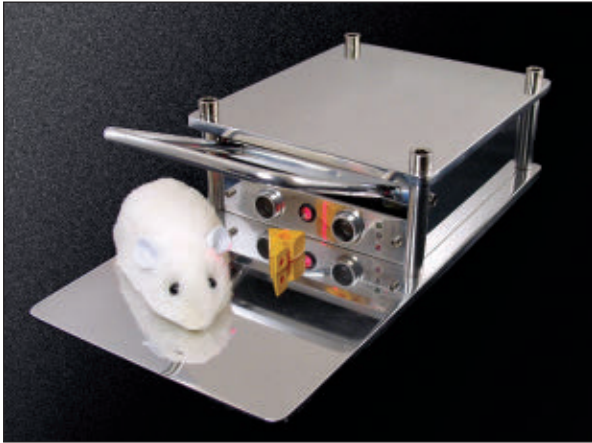
common bacteria can turn microgears when suspended in a solution, opening up the possibility of "bio-inspired dynamically adaptive materials for energy." The microgears — 380 microns long with slanted spokes — are placed in a solution along with the common aerobic bacteria *Bacillus subtilis*. The bacteria swim pretty much at random, but when they collide with the spokes of a gear, it turns them in a set direction. It takes a few hundred bacteria to turn each gear, and when multiple gears are connected (as in a clock), the bacteria will turn both gears in opposite directions, causing them to rotate in synchrony.

The gears' speed can be controlled by varying the amount of oxygen in the solution, reminiscent of how a carburetor throttle controls engine speed. It seems like it would take quite a few bacteria to drive anything heavier than a golf cart for an amoeba, but, according to ANL's Igor Aronson, "Our discovery demonstrates how microscopic swimming agents, such as bacteria or man-made nanorobots, in combination with hard materials, can constitute a 'smart material' which can dynamically alter its microstructures, repair damage, or power microdevices."



*Silhouettes of several gear designs that could be turned by *Bacillus subtilis* bacteria.*

Robotic Rodent Riddance



Telovation's Better Mousetrap primed for the kill.

In a fine tribute to Rube Goldberg and a stick in the eye to Ralph Waldo Emerson, Jake Easton of Telovation (www.telovation.com) has unveiled his robotic Better Mousetrap. What we're talking about is a gratuitously complex contraption that employs ultrasonic sensing, a high-speed pneumatic cylinder, a laser, a key switch, and other features to enhance the process of detecting and eliminating its target. You have to admit, though, that it's pretty attractive, with its 1/4 inch thick polished aluminum case, analog pressure gauge, and lighted armed/detect/fire indicators. The biggest drawback seems to be that it really can't tell the difference between a mouse and your schnauzer. Given that it packs a strike force of 102 lb (46 kg) at 40 PSI, it would be a good idea to locate it strategically to avoid accidents.

Body Fluid Bot for Labs

Not only is the collection and transport of specimens one of the least appetizing and somewhat hazardous jobs in the healthcare industry, it also accounts for somewhere



The SpecIMinder™ robotic specimen transporter, docked for a recharge.

between 46 and 68 percent of analytical errors and eats up as much as 50 percent of the average technician's time, according to prominent medical journals. To address the situation, CSS Robotics, Inc., and Swisslog Healthcare Solutions have put their heads together and come up with SpecIMinder™ — a bot designed to do the dirty work for us. Available for purchase or lease, the automated delivery cart claims to offer high-level security, timeliness, and reliability while being as simple to operate as a toaster.

Techs can select specimen destinations by simply pressing a button, and the bot will select the most efficient route, stop at all selected destinations, and return for duty when the job is done. It also finds its way back to the charging dock when not in use. According to the manufacturer, it can handle up to 50 lb (23 kg) of cargo per run, open doors, and (via wireless control) even operate elevators. For details, see www.ccsrobotics.com.

Keep Cleatus Out of Your Oral Meatus

In case you had been planning to buy one of the Fox Cleatus Robot action figures but haven't been able to find one, consider yourself lucky. A recent shipment from China was intercepted in Seattle by the US Customs and Border Patrol and tested for safety. Yep, you guessed it. They were found to contain 1,800 ppm of lead — six times the acceptable level as specified in the Federal Hazardous Substances Act. The 17,592 seized units were valued at \$96,000, or about \$5.46 each, considerably less than the common \$24.99 retail price.

If you already have one, you might try to avoid chewing on it, as lead poisoning causes a wide range of neurological, gastrointestinal, reproductive, renal, and other problems. **SV**

Cleatus action figures: Get the lead out!





GEER HEAD

by David Geer

Contact the author at geercom@windstream.net

Penbo For Girls, Prime-8 For Boys Walking Wonders Waddle, Roll, and Interact

Penbo is an intuitive emotive penguin robot — complete with baby — for girls. Prime-8 is a macho, aggressive gorilla robot for boys. Together, they represent hours of playtime for kids of all ages.

Bossa Nova Robotics is the eventual result of a \$20,000 DARPA project completed at Carnegie Mellon University. The immediate offspring of the project was a six legged robot named Rhex that walked on six rolling legs, each attached to a wheel. The legs hit the ground at alternating points to produce forward locomotion.

Today, two toy robots run on two such legs. Penbo the toy penguin walks by waddling. Prime-8 rolls on its arms.

Prime-8

A total of eight sensors and three motors educate and actuate Prime-8 the gorilla robot. Motors actuate the legs up and down and make the robot run on its legs. "The legs extend or retract to steer the robot," explains Sarjoun Skaff of Bossa Nova.

Sensors include two IR receivers, one emitter, and sound and touch sensors. The microphone together with the infrared enables touch detection. "The infrared knows when your hand is close to the robot and the microphone senses contact," Skaff says. The microphone enables the robot to hear and respond to the user's voice.

Prime-8 senses objects and motion. If the robot is not moving but detects that its infrared sensor has been triggered, then it

knows that something *else* is moving. If the infrared detects an object while the robot is moving, it knows that there is an object there and it moves to avoid it.

The robot uses a low-cost microcontroller with IC masking. The control software is burned into the chip to enable mass production at a lowered cost. Software that is not burned-in could become buggy or fail and have to be reinstalled.

Infrared is also a way for two Prime-8 robots to communicate and synchronize their behavior. "If you want to daisychain one in front of the other, you can synchronize the two and they will dance in tandem," says Skaff. The robots can also compete in laser tag.



Prime-8 from Bossa Nova Robotics is a gorilla robot that walks by rolling on its arms. It's technically made for kids eight to 12 years old.

Prime-8 Games

Prime-8 is capable of four games including Dodgeball, Race Running, Target Shooting, and Laser Tag. The user presses a button on the remote control to pan through the different game selections. At the Dodgeball selection, the robot will make a sound like falling bowling pins.

By hitting Prime-8 in his blue center chest plate in this game mode, the user can make it fall over. The robot will try to avoid being hit, however, and if it is struck somewhere else, it will only laugh.

In Race Running mode, the robot signifies the user is in the correct mode with a sound like an engine starting. When the user presses the "shoot" button, the robot will count down and then take off rolling on its arms. It can race another Prime-8 or the user. It moves about 0.8 meters per second.

In Target Shooting mode, the robot makes a sound like a rocket firing. The user simply installs the rocket launchers and rockets, and aims for the target (target cut-outs come with the robot). The user aims and shoots via remote control using the up, down, left, right, and shoot commands.

In Laser Tag mode, two Prime-8s can chase one another, target, and shoot lasers at each other which are recognized via infrared. To select a robot to control, one user must toggle to A and the other to B on the remote control. Each direct

hit scores six points.

The robot can also be made to tap dance as it responds to the sound of the user's hands clapping repeatedly. Tickle the robot's belly and it will break out in laughter.

Moods and Animations

Prime-8 has many moods and animations. When it is in a good mood, it can be a little rude but mostly it is compassionate. It will break dance, run in circles, and whistle.

When it is in the Gone Bananas mood/mode, it will pretend to swim on its back, lift virtual weights, stand on its head and stretch, and fire its lasers. To put the robot into Gone Bananas mode, simply wake it with a clap once it has fallen asleep.

Remote Control

The remote has 24 different controls to make the robot move back and forth, left and right, and go into turbo mode (fast!). One button is dedicated to animation and will cause the robot to enact one of 15 different sequences at random with each press.

One button makes Prime-8 autonomous, in which case it will run around on its own and sniff the furniture while avoiding objects or people. By pressing the demo button, the user can take the

The Penbo penguin robot from Bossa Nova Robotics is a friend for kids ages four to six.



Penbo's baby pops out from her belly. The baby becomes a remote control for optional RC use.



robot through all its capabilities automatically.

In guard mode, the robot will detect intruders and fire rockets at anyone who comes near.

Other control buttons enable programming the robot in unique sequences of movements and animations. "Simply record a sequence of commands and press play to execute," says Skaff.

The robot is available in the US for \$79.

Penbo

Penbo's action is possible because of three motors: two for leg motion and one for wing flapping, eye opening, and release of the baby ("bebe") egg from Penbo's stomach.

Penbo interacts with its environment based on input from seven sensors including infrared emitter-receivers, sound and touch detectors, egg detector, and baby detector. "Some of the sensors are inward-looking and some are outward-looking," describes Skaff. This means that some sensors study what is going on inside the robot, while others focus on what happens outside the robot.

The sound sensor enables the robot to hear the user and respond in its native toy penguin language — Penguinish — which is a combination of Penguin and English. A radar sensor enables Penbo to know the user's location so it can respond accordingly.

Some sensors are directed at object recognition and avoidance while a capacitive sensor in the toy penguin's head senses touch. If the penguin is in a good mood, a touch to its head triggers a happy response. In a bad mood, a touch will evoke a sad reply.

"When you stand Penbo on its head, it loves that and intuitively responds in a positive manner," says Skaff. Many of the robot's other functions are triggered by a head touch. By tapping the head twice, the user can make Penbo do a cha-cha dance.

Bebe has sensors too. These include an infrared signal to call out to Penbo along with its sound. Penbo hears through the sensor in its heart and communicates with the baby.

Penbo uses a processor with the software burned permanently into the silicon of the chip to enable low-cost mass production, according to Skaff. Bebe becomes a remote control when the user presses both wings at the same time for two seconds. "In RC mode, the user can make Penbo walk forward by pressing on Bebe's head, turn by pressing on the left or right wing, or dance by pressing both wings simultaneously," explains Skaff.

Penbo Games

Penbo can play five different games interactively including Tag, Hide and Seek, Peek-a-Boo, Marco Polo, and Musical Chairs.

Touch the heart twice and tap the head, and Penbo's Tag game becomes active. As Penbo takes off waddling away, the user must tap the head to stop it. As Bebe is also a remote control, the user can activate Penbo's Hide and Seek game by pressing Bebe's left wing (button). When Hide and Seek is activated, the robot turns in circles searching for Bebe.

The user participates by pressing Bebe's head to make it cry out "Mommy!" When Penbo hears its precious offspring, it waddles toward the sound. If Penbo locates Bebe again and hones in on its location, it becomes elated. If Penbo doesn't hear Bebe, it becomes disappointed. "If Penbo doesn't locate Bebe after three attempts, it exits game play," says Skaff.

In Peek-a-Boo game mode, the user dances while Penbo sings. When Penbo stops, the user must sit in front of Penbo quickly. "If you do and it senses you are there, you win," says Skaff.

With Marco Polo, Penbo again searches for its baby. Penbo spins around and calls out "Marco?" Bebe responds "Polo." "If the baby responds while the mother is facing her, Penbo waddles to her and opens her egg hatch for the baby to return," Skaff points out.

Penbo has another game called Mimic Me. In this game, Penbo makes a sound and the user must repeat it. Penbo picks up that sound with its sound detection sensor. If the sound the user makes is of the same pattern, Penbo responds happily.

Two Penbos are able to talk to each other or synchronize and sing a duet. Penbo retails for \$59 in the US.

Conclusion

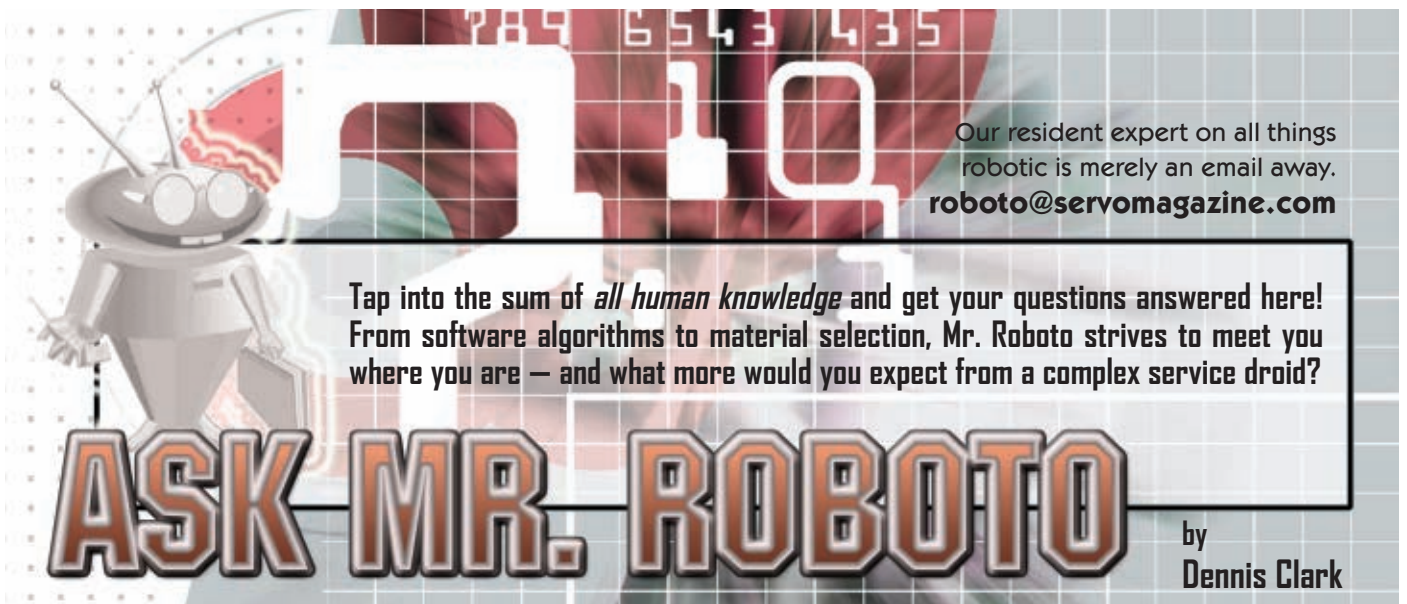
Both robots are unique additions to the world of animated, interactive play for kids. It won't be long before someone posts Prime-8 and Penbo, hacks soon enough. **SV**

Resources

Bossa Nova Robotics
www.bnrobotics.com

Prime-8
www.prime8robot.com

Penbo
www.penbo.com



*April marks my two year anniversary of writing this column. This month, I got a question that practically has me going full circle back to one of my earlier questions about one of my favorite topics: a **NON** Windows OS tool install. Without further reminiscing, let's get started.*

Q I use Linux for most everything and have started programming MCUs under Linux, but there are no good tutorials to be found on how to do this from scratch. The tutorials found are often about the GNU tool chain and its installation, but I'm looking for the whole thing.

Since you publish so many tutorials regarding programming (but they are all WinX based), I think it should be fun to publish one regarding the open-source world of OS.

If the tutorial could include the building of a tool chain and a makefile, and the use of any text editor based "hello world" example, it would be great! AVRDUDE is a great program to use under Linux/MAC/WIN and also deserves a place in this tutorial I ask for. Eclipse + plugin for AVR and a simple LED trick on a breadboard would do the trick.

A breadboard based tutorial is a cheap and good way to start regardless of which MCU or OS or programming language one uses.

— **Many thanks from Sweden and a pleased subscriber!**

A . Wow! That is a tall order! I see your point and I feel your pain! As a regular reader you have noticed that I am a Mac OS X person and rarely write about Windows programming or environments (the PIC being the exception, since it is currently Windows-only based). Way back in August '08, I wrote about setting up the avr-gcc tool chain with Eclipse and an AVR plugin on the

OS X platform. It took me some time to fiddle with everything but I have succeeded in getting the full avr-gcc on the newest Eclipse platform to work fine on Ubuntu 9.10 (Kharmic Koala). Following is my tutorial on how to accomplish this feat. First off, I'd like to tip my hat to the avr-gcc development crowd for their heroic work in creating and supporting this open source project. While I'm at it, I'd like to also thank the Eclipse gang, Avrdude development gang, and last but not least the developer of the AVR Eclipse plugin. The combination of all of these fantastic open source zealots has made our lives that much more interesting now that we have these tools!

I will detail here where to find the various parts of the AVR tool chain, Eclipse, and the AVR plugin. I'll show you how to install and configure it all, how to set up an AVR serial port programmer device in Linux, and how to create a tool in Eclipse to do the program download at the push of a button. For a project, I'll use a simple blinking LED program for a simple ATMEGA328 circuit. This will demonstrate a fully working system built from scratch on a common Linux distribution. (That should get you going with your AVR hobby!)

There should be a lot of useful information here since it's not as easy to set this up and configure it as it would be on the Macintosh OS X. It isn't super difficult either, so don't stop reading yet! We'll be getting the avr-gcc 4.3.3 tool chain and the Eclipse Galileo IDE, along with avrdude 5.8. This will get a little complicated, we'll use the "a picture is worth a thousand words" style of teaching with a lot of screenshots. Those of you that know your way around Linux can ignore my explanations of how to do things, and go directly to the good parts. This tutorial will be helpful for those who are just rusty or completely new to the Linux environment.



Figure 1. Get the avr-gcc tool chain.

Speaking of such, you'll notice that "sudo" is used a lot; this is a UNIX command that allows a user to do something at root, or superuser. The "su" stands for "SuperUser" and you can figure out what the "do" means. When you use sudo, the OS will ask you for the superuser password which you create when you install your OS.

Step Zero: Know where we started!

I downloaded and installed Ubuntu 9.10 (Karmic Koala) as my Linux distribution of choice. I chose Ubuntu because it seems to be the most popular strain of Linux I'd been hearing about lately. Here is where I got my it:

www.ubuntu.com/GetUbuntu/download

Just follow the instructions for downloading the ISO image and burning it to a CD-ROM. Then, boot from the CD-ROM you just made and follow the directions. Probably any Linux will do that supports the apt-get distribution tool – I've used this tool on Suze, Red Hat, and Ubuntu. So far, I like Ubuntu's setup the best, however, your mileage may vary.

Step One: Install the avr-gcc tool chain.

After you have installed and configured your

Ubuntu system (or if you already have a version of Linux that you like up and running), it is time to get the avr-gcc tool chain. On Linux, apt-get is your friend; we'll use it to install and set up the entire tool chain, largely auto-magically! Open a terminal window and type the following: `sudo apt-get install gcc-avr avr-libc avrdude librx-java` (Figure 1).

You'll need the librx-java if you want to play with the Arduino environment later on. I'm planning on doing a quick tutorial on that install in a later installment, so let's just drop it in too. The compiler is called avr-gcc, but for some reason the apt-get package is called gcc-avr – don't let that confuse you.

What makes apt-get so nice to work with is that it will download and build what you are installing. While it does that, it sorts out any dependencies that the package needs and downloads and builds those too. So, when you run apt-get, be ready to sit back for several minutes while files are found, sorted, downloaded, installed, and finally built.

This apt-get package will install the tool chain in `/usr/bin`. This would not have been my choice – I like to put things in their own little sandbox like `/usr/local/avr-gcc`, for instance – but I didn't want to fuss with apt-get options to move the install and perhaps mess up some path dependencies down-stream. So, I left it where it put it. To prove to yourself that all went well, type `avr-gcc -v` on your command line (letter case matters!); you'll see that you are running version 4.3.3. Likewise, type `avrdude -v` and it will report version 5.8. You're good to go ... on with the next step!

Step Two: Get Eclipse CDT for your projects.

Why do we want Eclipse? What does it do? Eclipse is a generic IDE (Integrated Development Environment) that is designed to work with a variety of programming languages. However, by itself, Eclipse is really nothing more than a customizable text editing program. If you add on plugin modules, then Eclipse becomes a whole lot more. The CDT version of Eclipse is customized for C and C++ programming so that it recognizes those types of syntax and can help you with a variety of editing enhancers. With the AVR plugin, Eclipse also handles building makefiles and generating the correct build flags for the avr-gcc command line compiler and linker. You can add Subversion or any other software versioning plugin supported, and you can debug your software with GDB, as well. Many different platforms use Eclipse as the IDE of choice. I'll show you how to

Figure 2. Get Eclipse from the source.

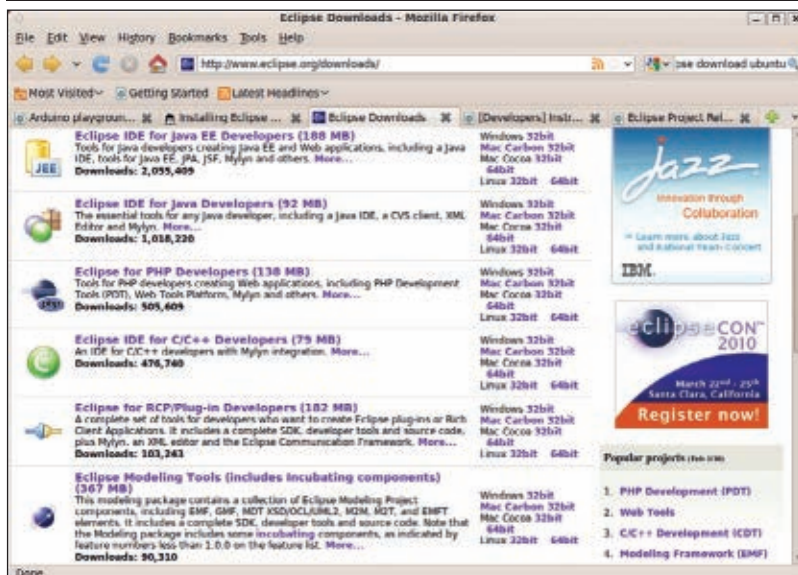




Figure 3. Use gunzip.



Figure 4. Uncompress the Eclipse install.

configure a tool in Eclipse to also program your AVR using avrdude. Eclipse is a very handy tool for a developer.

Eclipse comes in a lot of flavors for a lot of different programming languages, from java to pgp. We only want the C/C++ environment; just getting the CDT environment means a leaner program that will run faster, and will definitely download quicker. Go to www.eclipse.org/downloads and scroll down looking for *Eclipse IDE for C/C++ Developers*. When you find it, click on the *Linux 32bit* link to the right of the description (**Figure 2**). This link will take you to the actual download page. Click on the *Download* arrow and wait for it to come in. Firefox (as configured on my Ubuntu OS) put the download file in /tmp/eclipse-cpp-galileo-SR1-linux-gtk.tar.gz. Firefox may want to open the file in a tool that will uncompress it. **Don't** let it do that! We're going to put Eclipse where we want it. This file will dump it into /tmp if you just gunzip it, so simply save the file and we'll deal with it later.

Okay, now you have it, but it is in a *tarred* and *gzipped* file. It's not as pretty as getting an apt-get package, but apt-get wanted to download a fully loaded Java IDE with a bunch of things in it we didn't need or want for AVR development, so I went right to the Eclipse project to get an install. We can't run anything as a zip file, and we certainly don't want it in /tmp, so we're going to move the file and uncompress it in /usr/local/eclipse. First, gunzip the file in /tmp by typing `cd /tmp gunzip eclipse-cpp-galileo-SR2-linux-gtk.tar.gz` (**Figure 3**).

Now we just have a *tar* file. Let's create the directory we want Eclipse to live in. Type `sudo mkdir /usr/local/eclipse`. This will create our Eclipse directory. Now type `sudo mv /tmp/eclipse-cpp-galileo-SR2-linux-gtk.tar /usr/local/eclipse`. This command moves our *tar* file to our desired final location where we will uncompress it and create our Eclipse environment. Type `cd /usr/local/eclipse sudo tar xf eclipse-cpp-galileo-SR2-linux-gtk.tar` (**Figure 4**).

Congratulations! You now have avr-gcc and Eclipse installed. But wait! There's more! You will want Eclipse to know where your avr-gcc compiler is and

how to do AVR-type cool stuff. To do that, you will need the Eclipse AVR plugin. We can get this plugin directly from inside the Eclipse IDE. On to the next step ...

Step Three: Get the Eclipse AVR plugin installed.

This plugin handles all of the tedium of finding the compiler, deciding what compiler switches to use, and — *most* importantly — building the makefiles that build your AVR projects so you don't have to. Like everything else we've gotten so far, this too is an open source project. The first thing that we need to do is find the plugin so that we can tell Eclipse where to get it. Google helped me out with this task. Here are the download and install directions:

http://avr-eclipse.sourceforge.net/wiki/index.php/Plugin_Download.

This site is a little dated; we're going to be using Eclipse 3.5 (Galileo) but while the text around the software install process has changed a little, it is still recognizable enough that we can muddle through. If you have problems getting this to install gracefully (like I did at first), you can directly install the plugin as per the directions near the bottom of the page. Either way, it isn't too difficult. I'm going to show you how to install the plugin from within Eclipse.

In the upper left of your Ubuntu desktop, click on *Places*; draw down to *computer* and open that.



Figure 5. Ta da! Eclipse!

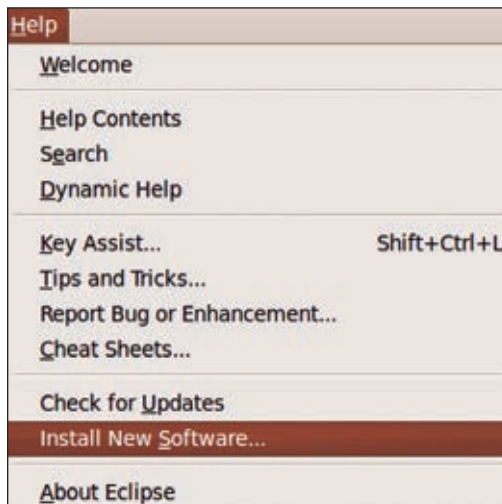


Figure 6. Add the new plugin software.

In this window, navigate to *filesystem-usr-local-eclipse*. You'll see your Eclipse executable here; double-click on it. Of course, if you are still in your command line terminal in the eclipse directory you can just type *eclipse* and get the same thing to happen. You'll start Eclipse up and see what is shown in **Figure 5**.

Here we go. Click on *Help-Install new Software* as shown in **Figure 6**. When the next window comes up, click on the *Add* button and you'll get **Figure 7**. Fill in these details:

Name: AVR Eclipse Plugin
 URL: <http://avr-eclipse.sourceforge.net/updatesite/>
 Then hit OK.

Now, click on the down arrow next to the

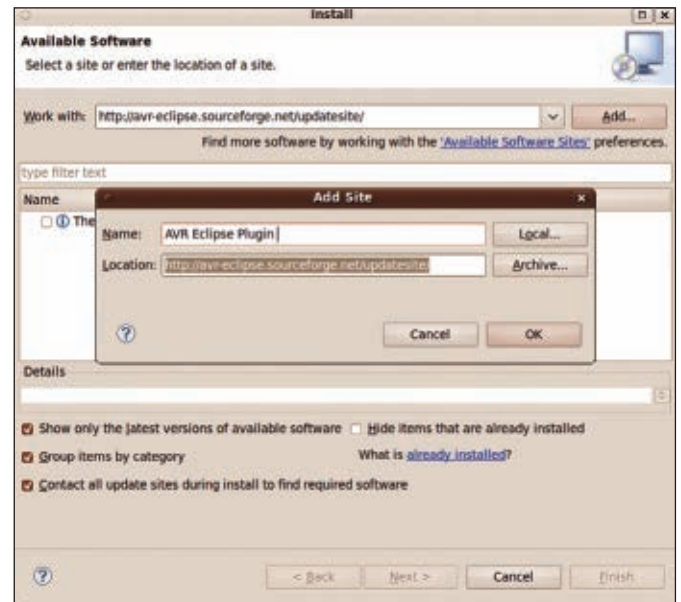


Figure 7. Details for the plugin.

Work with: window and select the AVR plugin entry. Type **AVR** into the window that has *type filter text* written in it. Now, select the *AVR Eclipse Plugin* entry that will appear in the main window and put a check into the checkbox next to the 2.3.1 version of the AVR plugin (**Figure 8**). I had to fiddle around with this window to make sure that I selected the correct fields, clicked the correct boxes, and wrote text in the correct windows. If you've done everything that I've said to do above, then the *Next >* button will become active and you can move on. If you didn't do everything, the button will not become active and you won't be able to move on and get the plugin installed. Go ahead and click *Next >*.

You will get a page that warns you about unsigned content. Don't worry, just click OK. You'll get a license page which you must accept or you won't get your plugin. Finally, you will get a

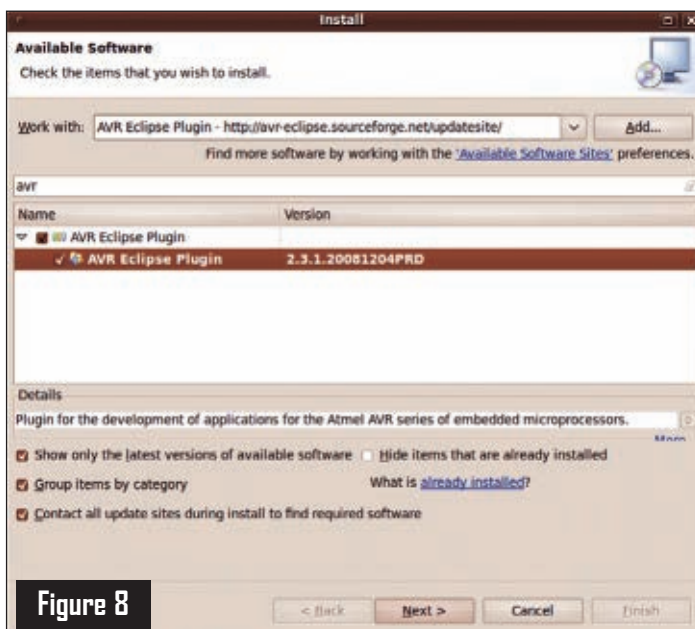


Figure 8

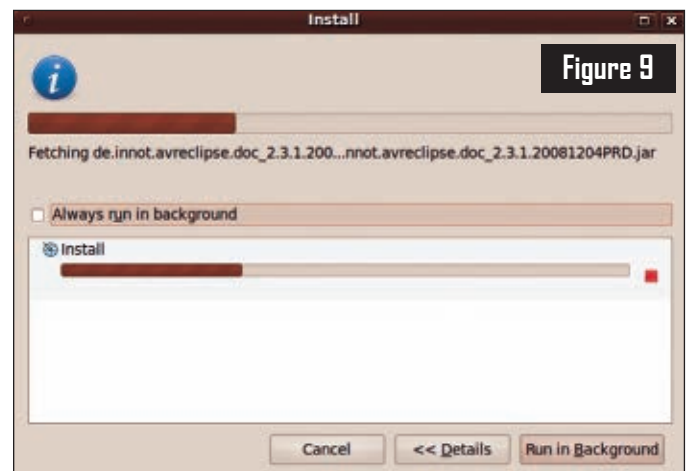
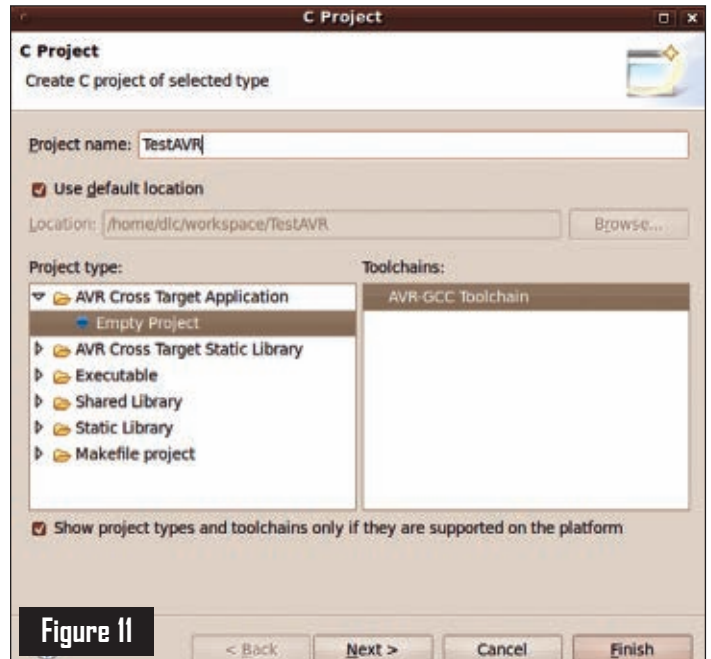
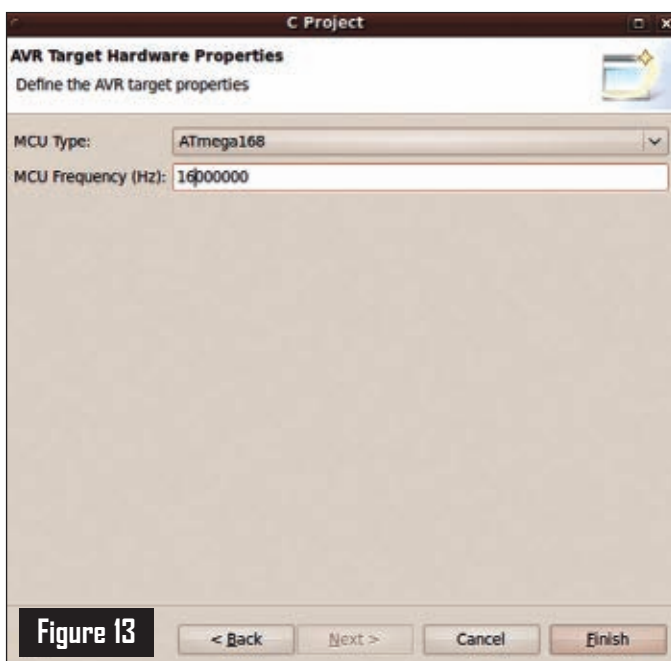
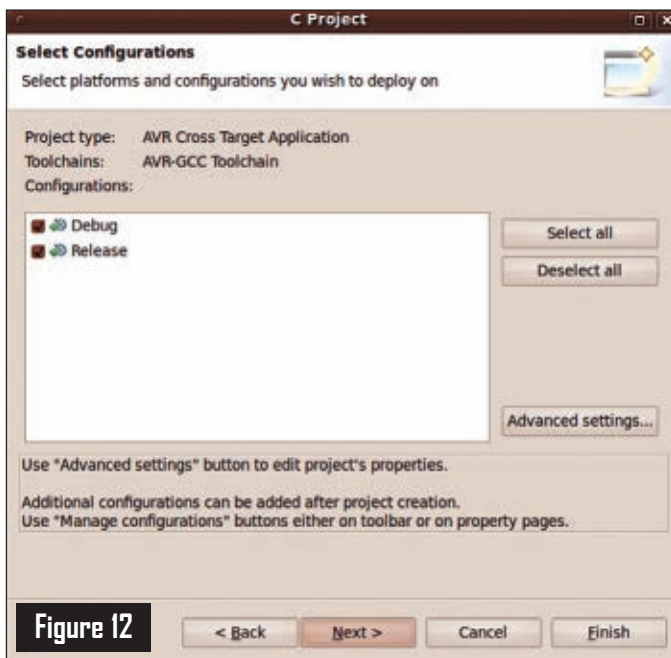
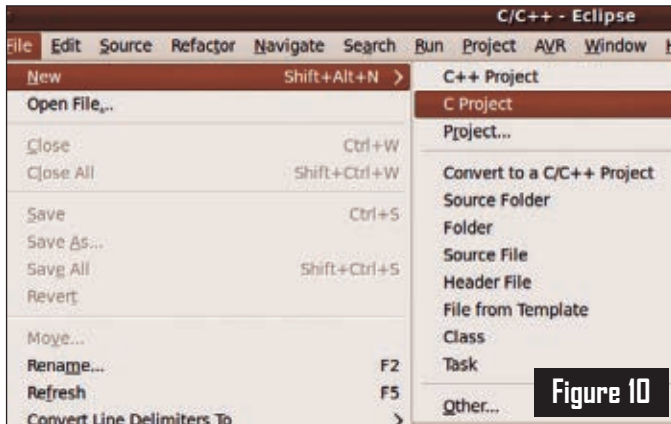


Figure 9



download progress window like the one shown in **Figure 9**. Now, you are home free! Wait for the software to be downloaded and installed; when a screen comes up that tells you that you should restart Eclipse, do so — it won't take long. Galileo is very fast; it'll be right back.

When you start Eclipse up, you'll get a window that asks where you want the *workspace* directory installed. I always take the default; it seems reasonable to me. You are now ready to try your first AVR program. Here are the steps for using the AVR plugin to set up a new AVR program.

- Select *File-New-C Project* (**Figure 10**).
- Give your project a name; in the left side window select *AVR Cross Target Application* and under that, *Empty Project*. In the right pane, you'll see *AVR-GCC Toolchain*. Hit *Next >* (**Figure 11**).
- Select if you want both a *Debug* and *Release* build option. I only use the *Release* option because I've yet to put the effort into getting GDB and AVaRice debuggers to work in the Eclipse environment. If you have done this, please let me know how because not only would I love to know, I'm sure many readers would too. For now, hit *Next >* again (**Figure 12**).
- Select your microcontroller. **Figure 13** shows the ATMEGA168; I'll be doing a demo LED blinker for the ATMEGA328p, so you might want to select that instead.

Select the clock speed. **Figure 13** shows 16 MHz which is my favorite clock speed. Hit *Finish* and you have now created a project – you just need to add files to it!

Step Four: Write an AVR gcc program and compile it.

This tutorial is about installing the tools to get going with the AVR microcontrollers, not about programming them. However, I feel you should have something to make sure everything works.

So, with that in mind, here is the embedded world's version of everybody's first program, "Hello World." It simply blinks an LED. Refer to **Listing 1** for this discussion.

Let's look at this simple program, function by function. First, **init()**. The DDRB, DDRC, and DDRD are the *Data Direction Register B/C/D*. These registers define whether an I/O pin is an input or an output. Each bit in the register corresponds to an I/O line. A logical '0' in a bit means that this line is an input; a '1' means it is an output. Notice that I set everything to be an output. This does two things:

First, it makes it easy to turn on an LED. The second reason comes from being a professional embedded engineer. We tend to make any unused I/O line an output so that we avoid generating "noise" on an input that can make a system unstable. The other way to avoid this noise is to tie said I/O lines to either Vcc or ground, and make the pin an input. Without generic boards, it makes more sense to make the pin an output if it's not being used. The ACSR register defines the use of the analog comparator module. If we don't turn this off, it will interfere with the digital functions on those I/O lines. Get into the habit of looking for a comparator on your micros and turning them off!

Lastly, we configure TCCR2A and TCCR2B. These are the Timer/Counter Configuration Registers for Timer2 in the ATMEGA168 and ATMEGA328. This is an eight-bit timer that can be used for PWM. In this case, we're configuring it as a simple timer. The 0x04 setting configures the

LISTING 1

```
/*
 * test.c
 *
 * Hello World for the embedded programmer.
 *
 * Created on: Feb 9, 2010
 * Author: dlc
 */

#include <avr/io.h>           //Takes care of ALL I/O definitions

void init(void)
{
    DDRB = 0xFF;              //all of port B is outputs
    DDRC = 0xFF;              //all of port C is outputs
    DDRD = 0xFE;              //RxD is an input, rest outs

    ACSR = 0x80;              //Turn off analog comparator

    TCCR2A = 0x00;            //Just a timer
    TCCR2B = 0x04;            //prescale 16MHz by 64 for 1.024mS/256
}

void waitms(unsigned int d)
/*
 * a ms timer routine.
 */
{
    int delay;

    for (delay=0; delay<d; delay++)
    {
        TCNT2 = 1;
        while(TCNT2);         //wait for a rollover
    }
}

int main(void)
{
    init();                   //Set up our processor

    PORTB = 0x02;             //Turn on LED on PORTB.B1


    while(1)                  //blink LED forever
    {
        PORTB = 0x00;
        waitms(500);
        PORTB = 0x02;
        waitms(200);
    }
}
```

prescaler to divide the system clock (16 MHz) by 64. When this is set up, a full count of 256 will *roll over* the TCNT2 register (where the counter is kept) every 1.024 ms. This is important to use as we'll see later.

Now let's look at the **waitms()** function. The only purpose for this function is to give us an easy to use timer. I *really* dislike a padded for/next loop for timing – that isn't timing, it is just wasting time so why guess? Set up a timer in your micro and use it. That's what the timer is there for. I'll be the first to admit that this timer function is crude, but it illustrates a way to use a microcontroller timer. In this module, we set TCNT2 = 1 and then sit in a while() loop until TCNT2 is zero. This is roughly 1 ms so we do this the number of times equal to the number of milliseconds we want to delay. Simple, huh? Use your timers; they are easy to use.

Finally, we get to the main program, coincidentally called **main()**. (Okay, so it isn't a coincidence.) When a compiler and linker create a C program executable, the linker sets things up so that the code starts executing at the first line in the main() function. In our little "Hello World" program, all that is happening in main() is that the LED is flashed after we initialize the micro. That's it – your first program! Type this in and hit the compile button (the one in the upper left of the icon bar that looks like a page with 1s and 0s on it). Or, you could navigate *Project-Build All*, or you could hit <control>B. Look down at the bottom of your Eclipse IDE and you'll see a window with some tabs on it. There are two we are very interested in: Console and Problems. Console shows the output of the compiler and linker, as well as any tool you use. Problems shows the compiler and linker errors with line numbers and is "hot." By hot I mean that if you double-click on the error or warning shown in the window under that tab, it will take you directly to the offending line in your program so that you can fix it. This is another neat feature of an IDE – hot links.

Next month, I'll show you how to make a quick ATMEGA168 or ATMEGA328 board and use avrdude to download the code to it. During that process, I'll show you how to set up avrdude as a tool in Eclipse so that all you have to do is press a button to have it automatically find and download the hex file to your board.


 Re: October 2008 Zombie.zip article; R/C receiver decoder. I found the referenced implementation of a "tank-like" RC decoder very elegant and would like to try it out myself.

1) Please elaborate on the minimum support PIC

hardware that I will need to implement your R/C decoder solution (i.e., what development platform in detail was used for the PIC 18F252).

- 2) Please elaborate on where I can obtain the support software, as well (i.e., CCS -PCH compiler).
- 3) Can this solution be ported to the PIC16 series or is the interrupt structure that you used the reason you chose to upgrade to the PIC18 series of processors?
- 4) Are there any other articles or reference information that will assist me in testing out this solution?

— Stephen

 I used MPLAB 8.43 (the latest), but any MPLAB in the 7.XX and 8.XX range will work just fine. I think though that the 8.XX MPLAB is more stable. I used the PIC18F252 because it gave me two PWM channels in a 28-pin part that wasn't very expensive. My programmer was the Microchip ICD2, but a PICkit 2 will work just fine and be a lot less expensive. Debugging will be much slower through the PICkit 2's serial interface, but it will work if you need it. MPLAB is free; the PICkit 2 is about \$40, and the ICD2 (now obsolete) was about \$170. I recommend the ICD3 for a bit more than the ICD2 was; it is way faster and more reliable!

The CCS PCH compiler is available at www.ccsinfo.com for about \$200. This is a decent small project compiler that is less expensive than other PIC C compilers. They are starting to get a bit more expensive these days, but they are still cheaper than the alternatives.

You can port this to the PIC16 family, but why bother? The parts aren't that much less expensive and – as you alluded to – the PIC18F family has the high speed interrupt capability that allows some shortcuts to be used for high priority interrupts, making the ISRs simpler to code.

I thought that you could Google and find lots of examples for this kind of project, but I found there really aren't any other easy-to-follow examples for doing what I did with an RC receiver and simple translation code to drive a tank style remote around.

That's A Wrap

I've gotten quite a spate of questions in this last month and nearly all of them are taking quite a bit of time to research! This tells me that there are a lot of folks out there needing help to get going, so remember, I'm here to provide answers to your questions. You know where to send them: roboto@servomagazine.com. **SV**

EVENTS

Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: <http://robots.net/rcfaq.html>

— R. Steven Rainwater

APRIL

- 8 Austrian Hexapod Championship**
FH Hagenberg, Austria
Six-legged robots dance and race their way to annual fame. Based on previous years, expect to see some interesting highlights on YouTube.
<http://fh-ooe.at/fh-oberoesterreich/aktuell/presse/fh-ooe-events>
- 8-10 BotsIQ (previously known as BattleBots IQ)**
Mare Island, Vallejo, CA
In this competition, students build remote controlled combat vehicles.
www.botsiq.org
- 10-11 Trinity College Fire Fighting Home Robot Contest**
Trinity College, Hartford, CT
In this event, autonomous robots must navigate a scale model of a house, search for a fire, and extinguish it.
www.trincoll.edu/events/robot
- 12-16 Alcabot-Hispabot**
University of Alcala, Madrid, Spain
In addition to regional Eurobot and Robocup Jr events, there will also be Sumo, Velocistas, and Rastreadores. My Spanish is a little rusty, but I'm guessing those last two are a robot racing event and a line following event.
www.depeca.uah.es/alcabot
- 13-15 DTU RoboCup**
Technical University of Denmark, Copenhagen, Denmark
This event includes varied courses including line and wall following.
www.robocup.dtu.dk
- 15-17 FIRST Robotics Competition**
Georgia Dome, Atlanta, GA
This is the big championship for FRC teams.
www.usfirst.org
- 15-17 National Robotics Challenge**
Marion, OH
This is the NRC Championship.
www.nationalroboticschallenge.org
- 16 Carnegie Mellon Mobot Races**
CMU, Pittsburgh, PA
This is the famous annual CMU autonomous mobile robot race.
www.cs.cmu.edu/~mobot
- 17 Istrobot**
Slovak University of Technology, Bratislava, Slovakia, EU
Events include IEEE Micromouse, mini Sumo, and freestyle.
www.robotics.sk
- 17 SparkFun Autonomous Vehicle Contest**
Boulder, CO
Autonomous ground and air robots must circumnavigate the SparkFun building.
http://sparkfun.com/commerce/product_info.php?products_id=9016
- 17 UC Davis Picnic Day Micromouse Contest**
University of California, Davis Campus, CA
The annual micromouse maze solving contest.
www.ece.ucdavis.edu/umouse
- 18 Robot-SM**
Västerås, Sweden
Sumo, mini Sumo, and a robot pentathlon.
www.robotsm.se

22-24 VEX Robotics World Championship
Dallas Convention Center, Dallas, TX
This is a really big world gathering of VEX teams for both autonomous and remote controlled competition.
www.vexrobotics.com/competition

23-25 RoboGames
San Mateo Fairgrounds, San Mateo, CA
BEAM, Mindstorms, combat, and just about every other form of autonomous and remote controlled robot competition you can imagine.
www.robogames.net

24 National Electronics Museum Robot Festival
National Electronics Museum, Linthicum, MD
This year's event is combined with a mini Maker Faire and will include robot contests, R2 builder demos, and regional hackerspace projects.
www.robotfest.com

24 Penn State Abington Fire Fighting Robot Contest
Penn State Abington, Abington, PA
In this event, autonomous robots must navigate a scale model of a house, search for a fire, and extinguish it.
www.ecsel.psu.edu/~avanzato/robots/contests

24 Penn State Abington Mini Grand Challenge
Penn State Abington, Abington, PA
Autonomous mobile robots navigate an outdoor course.
www.ecsel.psu.edu/~avanzato/robots/contests

24 The Tech Museum of Innovation's Annual Tech Challenge
San Jose, CA
This year's competition is called Space Junk.
<http://techchallenge.thetech.org>

24-25 Trenton Computer Festival Robotics Contest
The College of New Jersey Campus Ewing Township, NJ
In addition to computer and ham radio events, there are usually several autonomous robot contests. Previous contests included

maze navigation, precipice avoidance, and line following.

www.tcf-nj.org

MAY

2 Hawaii Underwater Robot Challenge
Richardson Pool, Pearl Harbor Naval Station, Honolulu, HI
ROV teams engage in timed, multitasking missions with tethered vehicles.
www.marinetech.org/rov_competition

3-8 ICRA Robot Challenge
Anchorage, AK
The challenge includes several events such as Mobile Microbotics, Mobile Manipulation, human-robot interaction, and a virtual manufacturing challenge.
<http://icra.wustl.edu>

7 SPURT
Technology Park Warnemunde Ltd., Rostock-Warnemunde, Germany
SPURT stands for School Projects Using Robot Techniques and the goal is to build robots that race each other on the SPURT track.
<http://spurt.uni-rostock.de>

8 Austrobot
University of Applied Science, Wels, Austria
Eurobot regional, this year's event is called "feed the world."
www.austrobot.info

Ardweeny. You'd think that's what all the big & mean Arduinos would call the smallest one. ...and you'd be right.

It's the smallest Arduino you can build yourself!

Mount all the parts on the Backpack PCB, and it takes up no more space on a breadboard than the included ATmega328 chip itself!

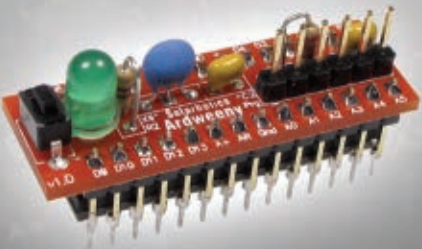
- Inexpensive
- Ideal for Breadboarding
- Fully Arduino compatible
- Simple to build!

SKU: KARDW
Price: \$9.95

SOLARBOTICS
www.solarbotics.com 1-866-276-2687

PS - But don't worry, he's like that small kid that gets picked on, but when the bully tries to push him, KAPOW! KARATE CHOP ACTION TAKES OFF HIS 44GS-USB PORT! TAKE THAT, SUKKA!

This image is to scale (...no, seriously!)





NEW PRODUCTS

ACCELEROMETERS

High-G Shock Accelerometers with Integral Electronics

Meggitt Sensing Systems has announced the launch of Endevco® model 7255A — a miniature, lightweight piezoelectric accelerometer with integral electronics. The rugged design and performance characteristics of this model make it ideal for use in near-field, high shock, and high level mechanical impact testing applications, where unwanted high frequency signals often mask critical low level, low frequency data.

Model 7255A incorporates a built-in mechanical filter, which blocks high frequency energy associated with near-field high shock, and protects the sensing element from



overstress. An internal electronic filter prevents saturation. The sensor features a two-wire system with integrated hybrid signal conditioner which transmits a low-impedance voltage output through the same cable that supplies required constant current power. Signal ground is connected to the inner case of the unit, acting as a shield. Both output and signal ground terminals are electrically isolated from the mounting surface. For lower level shock measurements, a high output version is available. Both versions include internal sensor assembly strain isolation which reduces strain input due to bending motion on the mounting surface (often found in high shock events).

Included with the 7255A are two small gauge, lightweight hook-up wires for use with the two solder pin terminal interface. The wire configuration is well-suited for this type of shock sensor, as wires do not exhibit the triboelectric effect under motion — a benefit which facilitates device mounting with minimal strain to both device and cable. The resultant strain minimization significantly reduces potential for zero shift, and ensures

NEW! Orangutan SVP-1284 Robot Controller

Item #1327 & #1328
\$99.95 and \$89.95



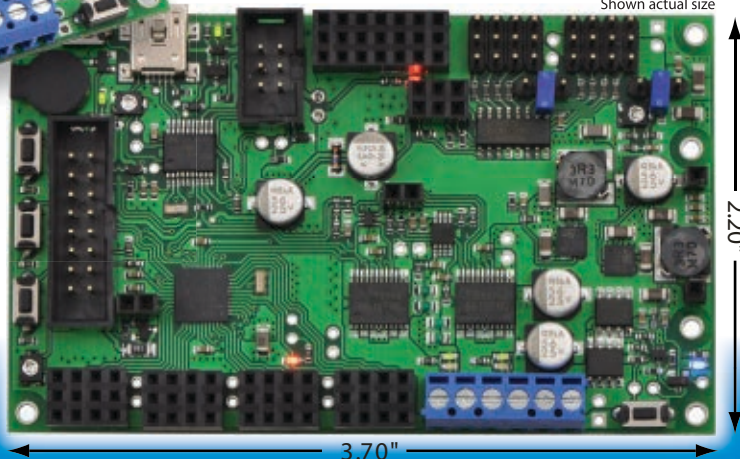
The **Orangutan SVP-1284** is available fully-assembled and as a kit.

The **Orangutan SVP-1284** lets you program your robot in C/C++ with free AVR development software. Use Pololu's AVR C/C++ library to easily make your **motors** go, **servos** turn, **LCD** print, and **buzzer** play - the Orangutan has all of these hardware features (and more) built right in!

Key Features:

- * **Wide input voltage range** - 6-13.5 V.
- * **C/C++ programmable microcontroller** - User-programmable Atmel ATmega1284PA AVR running at 20 MHz (128 KB flash, 16 KB SRAM, 4KB EEPROM).
- * **Built-in USB AVR ISP programmer** (USB A to mini-B cable included).
- * **Two bidirectional motor ports** - 2 A continuous/6 A maximum per channel.
- * **8 servo ports** - Use Pololu's AVR C/C++ library to control up to 8 servos!
- * **21 free I/O lines** - Up to 12 can be analog inputs, optional dual quadrature encoder inputs, two hardware UARTs.
- * **Removable LCD** - 16-character x 2-line, with backlight.
- * **Two step-down voltage regulators** - Each capable of supplying 3A.
- * **Other fun stuff** - Buzzer, 3 user pushbuttons, 2 user LEDs, and more!

Shown actual size



Pololu
Robotics & Electronics
3095 Patrick Lane #12, Las Vegas, NV 89120

Save 10%
with coupon code
SRV8SVP9

Learn more at www.pololu.com/svp or call 1-877-7-POLOLU

high reliability and performance.

For further information, please contact:

Meggitt
Sensing Systems

Website: www.endevco.com

ROBOT KITS

Seven New Educational Robot Kits

OWI, Inc.®, a leading supplier of educational RobotiKits™ (robot building kits for all levels of experience) has introduced.

After the award-winning success of OWI's 6-in-1 Educational Solar Kit (which allows young builders to assemble, disassemble, and reassemble six different types of solar-powered robots), the company now offers four new Mini Solar Kits. This award-winning line of solar-powered robot kits lets kids create their own fun, battery-free toy activated by a mini-solar panel. It's hands-on science without the carbon emission, as kids see how their toys speed up or slow



down depending on light intensity.

The new Mini Solar Kits include:

- T3 Transforming Solar Robot which transforms from tank to robot to scorpion with animated movement powered by sunshine or a halogen light.
- 3-in-1 Solar Stallion which pulls a chariot, flies like a Pegasus, or runs with the help of its trainer — powered by renewable solar energy.
- Solar System which will delight young astronomers as paintable planets revolve around their very own solar-powered "sun." The kit comes with six colors of opaque acrylic paint, plus brush.
- Solar Bullet Train which zooms with the help of its solar panel. Its 18 parts snap together and are secured with only four screws. Train enthusiasts of all ages will enjoy putting this together and watching it travel without rails.

There are also three new Aluminum Dinosaur Kits: T-Rex, Triceratops, and Stegosaurus. These were created following the success of OWI's Aluminum Bug Kits. While not solar-powered, these Jurassic buddies are some of the most lifelike dinosaurs young paleontologists will encounter. With soft pre-punched aluminum parts, screws, nuts, and assembly tools included, these are a great way to get kids interested in kit building.

For further information, please contact:

OWI, Inc.

Tel: 310 • 515 • 1900
Website: www.owirobot.com

THE ORIGINAL SINCE 1994
PCB-POOL
Beta LAYOUT

Servicing your complete PCB prototype needs :

- Low Cost - High Quality PCB Prototypes
- Easy online Ordering
- Full DRC included
- Lead-times from 24 hrs
- Optional Chemical Tin finish no extra cost

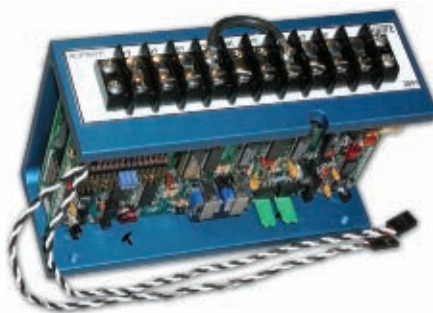
Watch "ur" PCB®
Follow the production of your PCB in **REALTIME**

FREE LASER STENCIL WITH ALL PROTOTYPE PCB ORDERS

email : sales@pcb-pool.com
Toll Free USA : 1 877 390 8541
www.pcb-pool.com

Beta LAYOUT

STEER WINNING ROBOTS WITHOUT SERVOS!



Perform proportional speed, direction, and steering with only two Radio/Control channels for vehicles using two separate brush-type electric motors mounted right and left with our **mixing RDFR dual speed control**. Used in many successful competitive robots. Single joystick operation: up goes straight ahead, down is reverse. Pure right or left twirls vehicle as motors turn opposite directions. In between stick positions completely proportional. Plugs in like a servo to your Futaba, JR, Hitec, or similar radio. Compatible with gyro steering stabilization. Various volt and amp sizes available. The RDFR47E 55V 75A per motor unit pictured above.
www.vantec.com

VANTEC

Order at
(888) 929-5055

bots IN BRIEF



CARE-O-BOT OPENS UP

Remember Care-O-bot 3 from Fraunhofer IPA — the lovable little robot that's missing a limb? Well, Fraunhofer IPA recently open-sourced their software.

The Care-O-bot 3 software platform was recently ported to ROS and is fully available as open source. You can now go to the **ROS.org** wiki and find documentation on the many software packages that they've released.

This is great news since developers will now be able to collaborate in making these robots do some truly amazing things — like accurate object recognition which is becoming increasingly prevalent in

consumer robotics as technologies develop. In Care-O-bot 3's case, objects are first scanned in three dimensions via a camera in the head. Once the bot has a 3D image saved, it can compare that saved image to live images from its camera to determine which object is which.

The Care-O-bot 3 stands 4'3" and has a single arm which is designed specifically to "relieve us of heavy, dirty, monotonous or irksome tasks." One of its primary functions is to serve drinks. Since many drinks come in varying containers, it can actually learn the shape of each bottle or can.

To teach it a newly-shaped container, simply place the bottle or can into its hand, and it will produce a 3D rendering which will be stored in its memory. There are also sensors in each finger which allow it to grip the container with the right amount of pressure, so as not to destroy your drink in the process of transporting it.

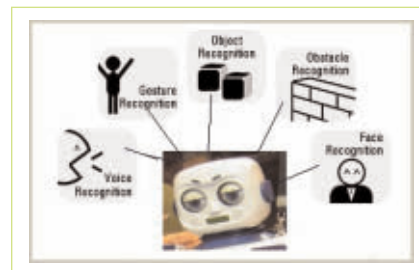
Currently, the Care-O-bot 3 takes orders via a touchscreen menu. However, there currently is a test unit which can receive audio commands, as well.



FUTURE LIES IN THE ATOM

FutureBots (www.futurebots.com/walk.htm) is advancing robotics to support the needs of mankind in space, healthcare, commercial, hazardous rescue activities, and chemical and biological disposal applications with the introduction of the ATOM-7xp humanoid robot.

ATOM — which stands for "Advanced Technology for Optimizing Motion" — was built by Dan Mathias, an electrical/mechanical/software engineer. ATOM-7xp has 49 degrees of freedom, is made of aluminum, titanium, and carbon fiber, and is all hand built in the USA in the FutureBots Labs by Dan. The bot has many high powered processor units running some of the world's most advanced operating systems. It also has advanced stereo vision system and an advanced walking system that imitates humans. Definitely check it out.



bots IN BRIEF

NEW NAVY SWEEPERS

You're looking at the Navy's newest recruits: (from left to right) CS3 Scooba Stevens, Chief Miles O'Brien, and ITSN Unger. No joke. That's what they've been named. The iRobot Scooba and Roombas are just part of an entire assemblage of robots who clean the floors on the USS Freedom, one of the newest and most expensive warships in the Navy.

Apparently, the robots are generally free to roam around the ship on their own. Crew members still have to do some sweeping, but the robots help keep things tidy on a day-to-day basis. Give it a couple years or maybe a decade, and the robots will be running the ship while humans do the sweeping.



GET IT IN GEAR

Developed by an educator, GEARS EDS hands-on engineering products are widely used from middle school through college and beyond. Their extensive curriculum delivers S.T.E.M. educational resources to aid teachers, and engage and inspire students. The website describes and illustrates The GEARS Invention and Design System™ which gives teachers the industrial strength tools they need to create world-class engineering and robotics challenges for their students in the comfort and convenience of their own classrooms. Videos of the system in use can be viewed and eight example projects that can be built with GEARS-IDS are explained.

An Education Tab on the site offers a full Curriculum Section, as well as Flash Programs demonstrating principles in various disciplines. Sample curriculum and lessons for the mechatronics kits Totally Trebuchet, The Pneumatics Trainer, and The Transmission Module can be downloaded for free.

Three different robotics levels offer programs for the novice through the graduate student. Electronic, pneumatic, and mechanical accessories each have their own sections, as well as gear head motors, batteries, and chargers. Users

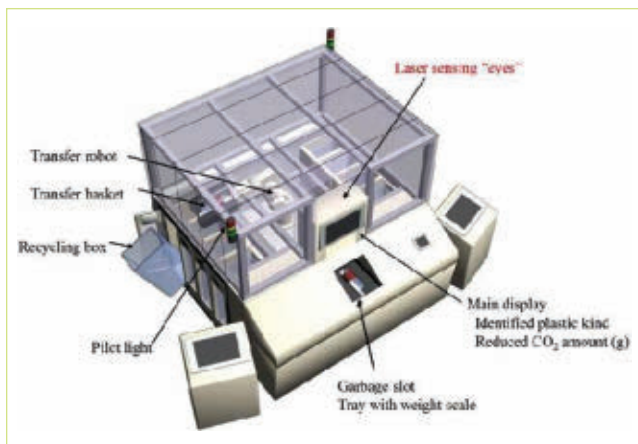
are encouraged to integrate their own sourced, fabricated, or salvaged components with GEARS products.

Useful links offer users directions to many other educational sites, as well as to GEARS partners.

Be sure to visit the easy-to-navigate site at www.gearseds.com.

A screenshot of the GEARS Educational Systems, LLC website. The header features the company name and navigation links: Home, Site Map, Contact, Robust Programs That Teach Science, Technology, Engineering, and Math, Hands-on Engineering, and Educational Resources. Below the header is a navigation bar with links: Who We Are, Our Products, Example Projects, Education, CAD, and Support. The main content area is divided into several sections: Gears EDS, The Gears Invention & Design System™, What's New, Mechatronics Training Kits (Totally Trebuchet, Pneumatics Trainer, Transmission Module), Robotics Programs (Starter Robot, Heavy Metal Chassis, Surface Mobility Platform), and a section for 8PC - This Platform. Each section includes a brief description and a small image of the product or project. The footer contains copyright information and contact details.

Cool tidbits and interesting info herein mainly provided by Evan Ackerman at www.botjunkie.com, but also www.robotsnob.com and other places.



WE TAKE PLASTIC

Despite the fact that plastics are used everywhere — especially in products which are intended to be disposable — the recycling rate for plastics compared to metal, paper, and glass is abysmal. It's somewhere around 6% which means 94% of potentially recyclable plastics end up in landfills or in the stomachs of our sea friends. The problem is that it's a gigantic hassle to do the obligatory pre-recycle sort with plastics. There are recycle numbers stamped on most plastic containers, but plastics with the same number often can't actually be recycled together. Basically, it's a mess.

Mitsubishi Electric Engineering Corp and Osaka University are tackling this problem with a robot, since we humans have proven

ourselves to be fairly worthless at plastic recycling. As you might expect, the robot has some capabilities that we don't — namely five lasers of different wavelengths that can be used to determine the reflectivity index (and therefore the composition) of most plastics. The robot will automatically sort plastics into six different types which takes the hard work of manual identification out of the recycling process.

There aren't specific details on how much plastic one of these robots can actually sort through every day, but there is a commercial version in the works which should be available (at some point) for around \$55K. At that price, it's not going to be resource neutral anytime soon, but at least it will make you feel better about your recycling habits. And the sea turtles will thank you.



AP CIRCUITS
PCB Fabrication Since 1984

As low as...

\$9.95
each!

Two Boards
Two Layers
Two Masks
One Legend

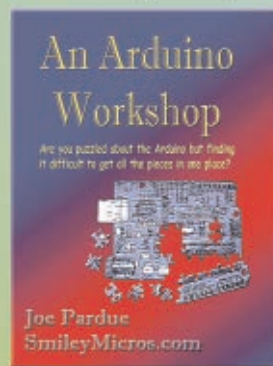
Unmasked boards ship next day!

www.apcircuits.com



Puzzled by the Arduino?

Based on the Nuts&Volts Smiley's Workshop,
this set gives you all the pieces you need!



Book \$44.95



Kit \$84.95

Book and Kit Combo \$124.95

For more info on this and other great combos
please visit:

<http://store.nutsvolts.com>
or call (800) 783-4624

ALL ELECTRONICS

C O R P O R A T I O N

THOUSANDS OF ELECTRONIC
PARTS AND SUPPLIES

VISIT OUR ONLINE STORE AT
www.allelectronics.com

WALL TRANSFORMERS, ALARMS,
FUSES, CABLE TIES, RELAYS, OPTO
ELECTRONICS, KNOBS, VIDEO
ACCESSORIES, SIRENS, SOLDER
ACCESSORIES, MOTORS, DIODES,
HEAT SINKS, CAPACITORS, CHOKES,
TOOLS, FASTENERS, TERMINAL
STRIPS, CRIMP CONNECTORS,
L.E.D.S., DISPLAYS, FANS, BREAD-
BOARDS, RESISTORS, SOLAR CELLS,
BUZZERS, BATTERIES, MAGNETS,
CAMERAS, DC-DC CONVERTERS,
HEADPHONES, LAMPS, PANEL
METERS, SWITCHES, SPEAKERS,
PELTIER DEVICES, and much more....

ORDER TOLL FREE
1-800-826-5432

Ask for our FREE 96 page catalog

iTelegraph



Send Morse code to your
friends using your vintage
telegraph set or built-in
sounder over the Internet!

- * **Arduino compatible**
- * **Open source**
- * **Hack it for other uses!**

For kits, motor controllers,
parts, Arduinos, shields
and more, visit:

www.criticalvelocity.com/itg

Firgelli

www.firgelli.com

LINEAR SERVOS



L12-R Linear Servo

Direct replacement for regular rotary servos
Standard 3 wire connectors
Compatible with most R/C receivers
1-2ms PWM control signal, 6v power
1", 2" & 4" strokes
3 - 10 lbs. force ranges
1/4" - 1" per second speed ranges
Options include Limit Switches and
Position Feedback

L12-NXT Linear Servo

Designed for LEGO Mindstorms NXT[®]
Plugs directly into your NXT Brick
NXT-G Block available for download
Can be used with Technic and PF
Max Speed: 1/2" per sec.
Pushes up to 5lbs.
2" & 4" strokes



PQ12 Linear Actuator

Miniature Linear Motion Devices
6 or 12 Volts, 3/4" Stroke
Up to 5 lbs force.
Integrated position feedback or
Limit switches at end of stroke
External position control avail.



Available Now At
www.firgelli.com

Robotics Showcase

Beginner Robotic Kits



Only \$49.95

bwsciencelabs.com/virus

**Finally! Full
motion control
for differential
drive robots!**



WheelCommander™

- Closed-loop position, angle, velocity & rotation rate
- Real world measurement units
- I2C and RS232 auto-switching interface
- WheelWatcher™ compatible
- Windows API & PID tuning wizard
- Examples for common robot controllers

NU-BOTICS™
www.nubotics.com

Recycling & Remarketing High Technology
WEIRDSTUFF®
WAREHOUSE

Software, Computers, Electronics, Equipment, Doo-hickies

**384 W. Caribbean Dr.
Sunnyvale, CA 94089**

Mon-Sat: 9:30-6:00 Sun: 11:00-5:00

(408)743-5650 Store x324

**WE BUY AND SELL EXCESS
& OBSOLETE INVENTORIES!**

FREE COMPUTER RECYCLING

We recycle computers, monitors, and
electronic equipment. M-Sat 9:30-4:00



WEEKLY SEALED BID SALE AUCTION
Hi-tech items, electronics
test equipment, and more!

GIANT AS-IS SECTION
10,000 sq. ft. of computers, electronics,
software, doo-hickies, cables, and more!



also check out our...

eBay Store
stores.ebay.com/WeirdStuff-Inc

WWW.WEIRDSTUFF.COM

COMBAT ZONE

Featured This Month:

Features

28 BUILD REPORT:

Crossfire – From Broken Robot to Breaking Robots

by James Baker

30 MANUFACTURING:

Keeping Your Stuff Together

by Nick Martin

33 PARTS IS PARTS:

FingerTech Robotics' Spark Motor Series

by Kevin Berry

33 *How Not to Die. I Mean, Build a Test Box*

by Matthew Spurk

35 *Combat Zone's Greatest Hits*

by Kevin Berry

Events

32 *Jan/Feb 2010 Results and Apr/May 2010 Upcoming Events*

BUILD REPORT:

Crossfire – From Broken Robot to Breaking Robots

● by James Baker

Metal Fatigue

A veteran of televised robotic combat, the heavyweight robot "Edgehog" was still on frontline service when it entered the 2008 UK live event season. Continuously upgraded over seven years of combat with better armor, Edgehog was now as tough as it could get without breaching the 100 kg weight limit. Unfortunately, the 2008 season and the fantastically powerful robots it involved proved to be the end for this plucky old machine.

In a successive series of events, Edgehog was extensively



Crossfire: well armored, well armed.

damaged by a powerful hammer, repeatedly pierced by hydraulic claw weapons, and pounded by pneumatic axes. The robot was effectively destroyed three times in as many months. After each event, the armor was patched and welded back together, the

Internally, Crossfire is very simple.



The 5/3 valve controls the ram.





The axe linkage is unique.

chassis cut, reinforced, and rebuilt, and the damaged internals repaired or replaced. It was getting expensive, time consuming, and frustrating. More importantly, however, the space frame chassis was so patched and welded that it was cracking in almost every fight. It was time to build either a completely new chassis for Edgehog, or build a successor.

Heavy Weapon

In the good old days, axes could pierce armor. By utilizing a sharp, lightweight axe head, a large pneumatic ram and an acceleration linkage, holes could be punched in the steel, aluminum, or thin polycarbonate skin of the opponent quite regularly. More recently, the robots have been armored in thick titanium, heavy



The twin pressure bottles and large ram sit under the robot.

steel wear-plate, and thick polycarbonate. To break through such material and get to the internal components was almost impossible.

The solution chosen for our new robot "Crossfire" was a heavy, blunt trauma weapon; an 11 kg bludgeon that would damage the enemy, bend the armor supports, and twist the chassis. This medieval looking appendage was to be swung through a unique, innovative linkage from the 100 mm pneumatic ram.

The linkage acts like a gearbox, taking advantage of the huge linear forces available from the ram (which runs at only 200 psi, but is capable and tested to 1,000 psi should the motivation ever arise) by way of an unequal length swing arm and custom-made spur gear. The result is that the 11 kg bludgeon swings at one and a half times the speed of the ram, through 180 degrees of motion, lifting the whole machine into the air, and hitting with enough energy to bend half inch steel plate. It also has more than enough force to right the whole robot should it be turned over during battle.

Heavy Armor

From the lessons learned with its predecessor, Crossfire had to be tough. So tough in fact, that it has never been externally damaged since entering its first combat event in 2009. The outer skin is made of a specially selected high carbon content steel wear-plate, shaped and welded in such a way as to deflect, resist, and survive any current weapon in the UK.



Testing the weapon.

The hard outer shell is mounted to the curved internal spine with heavy-duty brackets and bolts, welded reinforcing beams, and struts. All internal components are shockproof, and flexibly mounted where possible. The result is a rock solid machine that has yet to be scratched.

Carefully Selected Components

Having spent several years updating, upgrading, repairing, modifying, and (in some cases) destroying our motors, speed controllers, and batteries, we have a clear understanding of what works and what is reliable in UK-based combat. Unfortunately for us (and our national pride), what we have found over these many years of testing is ... buy American! The motors and gearboxes are manufactured in the USA for heavy-duty electric wheelchairs. Controlling these 24 volt gear-motors is a single Robot Power Sidewinder; with a pair of used 36 volt NiCad Battle-packs



Scars of a difficult season.



Edgehog was badly damaged.

bought from a competitor in the US via eBay. This set-up is sufficiently powerful, fast, and reliable, but also extremely cost-effective. The 40 MHz radio control system used is cheap and reliable, although we would now buy a 2.4 GHz system instead if starting all over again, as it offers more safety and convenience by effectively eliminating the chances of interference or loss of control.

The storage of the liquid CO₂ used in the pneumatic system is critical, so a certified fire extinguisher is used for the main bottle and an identical extinguisher (painted black) is used for the buffer tank which allows the use of the CO₂ full bottle pressure to be used if required; a change of pipe work is the only modification needed. The choice of pressure tanks used is also very cost-effective. A standard, high-pressure welding regulator is perfectly capable of handling the reduction to 200 psi. Likewise, the perfectly capable Electronize weapon switcher and 5/3 solenoid valve are a neat and inexpensive solution that has the option of going to 1,000 psi with little modification.

Finally, we have incorporated a simple but tidy charging system that allows us to remove our safety link and charge the batteries without turning a single bolt. We use jack plugs hidden under the arm of the bludgeon.

Development

After competing in several live events in 2009, a number of updates were needed. Firstly, the thick wall steel box section used in the arm was bending under the huge forces. The arm was reinforced along its full length with steel plate, which also added a little more weight to the bludgeon. This solved the bending problem in Crossfire, and caused more bending problems in the enemy robots (which is a nice win-win situation for us). The gears in the original boxes stripped unexpectedly, so larger versions of the same gear-motor were fitted and have been reliable to date.

The cause of the failure was traced to bolts coming loose in the gear housing, allowing the drive worm to wobble around and destroy the teeth on the drive gear. This will not happen again as all bolts now use thread lock. (See Nick Martin's article on thread locking in this issue of Combat Zone).

Finally, the robot carries much of its weight relatively high up, so its center of gravity is higher than on Edgehog. This means that it does get flipped over a little easier and often takes a few attempts to right itself. This can be solved with some welded fins strategically placed on the outer shell, but the driver is happy with the look of the robot and prefers

to leave the robot as it is for now. We may add the fins for important competitions in the future.

Does It Work?

Absolutely! All design targets have been met. The machine has been reliable, cost-effective, low maintenance, and is easy to turn around for the next fight. The armor is just as impervious as we hoped for, and the weapon is devastating against many opponents — often immobilizing them by bending the chassis enough to lock up the wheels or crush internal components.

The inspiration for the paint job arose because we often wear combat jackets at events which has become an unofficial team uniform. The robot looks great in the arena in contrast to robots in primary colors or shiny metallic, so Crossfire has proven to be the perfect successor to and companion of our veteran Edgehog, who has still not retired and will be fighting again in 2010. **SV**

Resources

www.robotpower.com
www.battlepacks.com
www.electronize.co.uk
www.futaba.com
www.xbotz.com

MANUFACTURING: Keeping Your Stuff Together

● by Nick Martin

Since screws were invented over 2,000 years ago, mankind has struggled to keep them from coming undone. In the past 100 years, many advances have been made, yet screws still loosen and no single solution works universally. This article will examine what is available and what solutions work best in combat bots.

How Threads Loosen

Nuts and screws rely on friction between the threads to prevent loosening. We create the friction by tightening the parts until they are under tension. Thread loosening has several causes:

- Temperature changes cause the

expansion and contraction of fasteners and the bot's frame, which reduces bolt tension and clamping force in the assembly.

- Self-loosening, which is caused by a sliding motion between the fastener and the part it is clamping.
- Excess loading can stretch bolts or compress the parts they are clamping and reduce the tension between them.

FIGURE 1. Split, star, and Bellville washers.



Unfortunately for us, all these causes are abundant in combat bots! Here are the various ways to deal with them.

Spring Washers

Spring washers are the oldest and least reliable way to secure a bolt. They all work by adding friction and tension between the bolt and substrate. Unfortunately, it only takes from 0.25 to two revolutions of the bolt before the spring washer loses its tension and becomes useless. There are many variations with different benefits.

Bellville washers are conical washers used for maintaining a uniform tension under a fastener, and do not provide any real locking function. Until they are completely flattened, they act as a spring; their main use in bots is as clutch springs or as shock absorbers.

Wedge washers have two parts that act as a ratchet and require more force to release than to tighten. They are probably the most effective type of locking washer, but have the drawbacks of extra weight, volume, and extra parts to lose.

Star washers have a number of sharp edges that provide some locking action, but cause surface damage. Their main use is on lower stress applications such as electrical connections where a liquid thread locker is unsuitable.

Split washers have been widely discredited as useless. Both NASA and military tests show that when compressed, they act the same as plain flat washers and have little holding power.

FIGURE 2. Nylon and all-metal lock nuts.



Lock Nuts

Lock nuts are a step up from spring washers with several important advantages. There is one less part to lose and, most importantly, the locking action does not rely on tension between the nut and the frame to stay tight. This way, they resist loosening right up to the point where they come off the end of the screw. The most common lock nuts have a nylon insert and are effective up to about 250°F. The all-metal variety is more expensive and works at high temperatures — this type is ideal for flamethrower bots.

The only real problem with lock nuts is that they only work where you have a through hole or an externally threaded part.

Liquid Thread Locking

The best known brand is Loctite. However, there are others and you can even use hot-melt glue in an emergency. The pros and cons of liquid thread lockers are:

- One size fits all. You will not need to stock different sizes and types of washers and lock nuts. At most, you will need two or three different grades of Loctite.
- It's hard to lose Loctite compared to small washers, although you

TABLE 1. Curing times for Loctite products.

Product	50% Strength (hours)	100% strength (hours)
243	0.5	3
248	0.25	24 (75% @ 1 hour)
262	0.75 (0.25 @ 100 °F)	8
246	0.2 (0.1 @ 100 °F)	24 (75% @ 2 hours)

FIGURE 3. Liquid and solid stick Loctite.



may find other builders lining up to borrow it.

- Loctite needs time to cure, making it less suitable for screws that are constantly removed during events.
- Loctite is an anaerobic glue, which means it cures in the absence of air. Loctite stays liquid until you screw the parts together, after which it will begin to harden. All varieties need some curing time to reach full strength, as shown in **Table 1**. Allow at least enough time for 50% strength to be reached.
- Loctite cure times are at 70°F with steel parts. Raising the temperature above 100°F will shorten cure times, while using less reactive metals like stainless steel will lengthen curing time and give lower holding power.
- Loctite products all lose strength at higher temperatures; 243 loses 50% of its holding strength at 210°F, while 262 is down to 50% strength at 300°F.

TABLE 2. Loctite selection chart.

Product	Strength	Cure Time	Oil Tolerant	Surface Tolerant	Temp. Rating
242	Medium	Medium			300° F
243	Medium	Medium	X	X	300° F
246	Medium	Fast			450° F
248	Medium	Medium - fast			300° F
262	High	Slow			300° F
266	High	Fast	X	X	300° F
268	High	Slow			300° F
2760	Very high	Medium		X	300° F

Selecting the Right Product

Loctite makes a bewildering array of products. **Table 2** lists some of the most suitable types for robots. My top recommendations are highlighted; they provide the best combination of features you need in a combat bot.

Using Loctite

Make sure that both parts are as clean and de-greased as possible. Cleaning with isopropyl alcohol or a similar solvent makes a huge difference, even with the grease-tolerant products like 243.

For screws secured with a nut, you only need to apply Loctite to the screw where the nut will be tightened. I like to assemble the screw in the part first so that I can see exactly where to apply the Loctite. This helps prevent the screw hole from gumming up with old Loctite.

For screws that are inserted in blind threaded holes, you need to apply Loctite to both the screw and the hole. Otherwise, air pressure will force much of the Loctite out of the hole as the screw is inserted. I find the end of a cable tie is perfect for distributing Loctite in small holes.

For fasteners where BOTH parts are made of inactive metals such as aluminum, stainless steel, magnesium, zinc, black oxide, cadmium, or

titanium, you need to use Loctite 7649 spray or liquid primer. The primer will reduce curing time and provide more holding power. For surfaces where one of the areas is brass, copper, bronze, iron, steel, or nickel, you don't need to use primer unless a short curing time is important.

Removing Loctited Screws

Parts with low and medium strength Loctite generally only need leverage to remove them. Parts with high strength Loctite are hard to shift. The recommended procedure is to heat the parts to 480°F for several minutes before trying to separate them.

Once the screw is removed, you can reuse it simply by scraping off the hardened Loctite and applying a fresh coat. The old Loctite can be tough to remove (if it isn't, you are doing something wrong!). A wire brush will get most of the material off, while a small triangular file will clean up tough spots. If you are really pressed for time in the pits, apply a small amount of Loctite over the hardened material; it is usually enough to get you through one more match and because the fresh Loctite is in a thin layer, it will cure faster.

Tips and Tricks

Clean new screws. Almost all

new nuts and screws have oil residue on them. A quick wipe with solvent makes a huge difference to Loctite's holding power!

Use high strength Loctite on small screws. For small screws with limited thread engagement, Loctite 266 is a good choice. The screws that secure 550 size motors are a prime example.

Heat parts for fast curing. Heating the head of a screw or a nut with a soldering iron or a hot air gun will dramatically shorten the cure time. Be careful not to overheat the parts; mildly hot to the touch for a few minutes is enough to get you ready for the grand finale.

Use liquid Loctite in the workshop and Loctite sticks at events. The solid stick types 248 and 268 are very practical in the pits; they can't leak, are less messy, and stay where you apply them. The only liquid Loctite I would take to an event is 246 for its fast cure times.

The 50 ml size Loctite containers have a nifty new snap-on cap and no-drip tip; ideal for the workshop. **SV**

Further Reading

If you want to know more about threaded joints, these two links will provide more than enough info:

<http://gltrs.grc.nasa.gov/reports/1990/RP-1228.pdf>

www.boltscience.com/pages/info.htm

EVENTS

Completed and Upcoming Events

Completed Events: Jan 10 to Feb 10, 2010

Kilobots XVI @ SPECTRUM was held January 16th in Saskatoon, Saskatchewan, Canada; www.kilobots.com.



Upcoming Events for April and May

RoboGames 2010 will be held April 23rd – 25th in San Mateo, California; www.robogames.net.



HORD Spring 2010 will be held on May 15th in Brecksville,

Ohio, presented by the Ohio Robot Club; www.ohiorobotclub.com.



Maker Faire Bot Gauntlet will be held in San Mateo, California on May 22nd and 23rd, presented by California Insect Bots; <http://calbugs.com>. **SV**

PARTS IS PARTS: FingerTech Robotics' Spark Motor Series

● by Kevin Berry

FingerTech Robotics (www.fingertechrobotics.com) has a new line of motors. Following my policy of only using this space to feature combat tested products, I delayed reviewing them until the head of FingerTech, Kurtis Wanner, had some combat data under his belt. Kurtis assures me, "January's Kilobots XVI and the University of Saskatchewan's Sumo competition put almost 50 of these motors through their paces. There were only a couple mechanical failures that weren't caused by direct combat damage or overvolting the motors." Since then, FingerTech has instituted a 100% testing policy for each motor, under load, before shipment.

Given that, I'm pleased to give some details about the specs and the design process. The Spark series has an extra-long 3 mm shaft, which is triple-supported internally so that wheels can be mounted directly to it. It uses the Mabuchi FK-050 motor, with nine different gear ratios (20, 35, 50, 63, 86, 115, 150, 250, and 360:1). Weight is 28 grams (one ounce). Cost is around \$18 US.

The Mabuchi motors will run on 4.5 to 22 volts, recognizing that anything above about seven volts is going to have a shorter lifespan. The

assembly is a little over 1.5 inches long and 0.6 inches in diameter. The shaft adds up to another 1.5".

Torque specs are shown in **Figure 1**.

Fingertech worked with the manufacturer to battle-harden the gearbox. **Figure 2** shows a beta version of the added support plate. According to Kurtis, "the indicated plate did not exist in their standard gearhead. The main shaft has one large gear pressed on, but also another gear that spins freely on it. Without the new plate, the thin shaft would snap between the two gears when any large sideload was placed on the shaft. Now it is supported by: 1) the faceplate bushing; 2) the new support plate; and 3) the old

Spark2 Original.



carrier (end of the shaft)."

Responding to my questions about testing and toughness, Kurtis supplied the photographic evidence (**Figure 3**) of how effective the three point support system is. He says, "This is a test I did with one of my modified motors; 20 lbs hanging off the end of the 1.5" shaft. The motor ran afterward without even a hiccup."

Even though I personally haven't used these motors yet, I'd say the jury is in. Good 'nuff! **SV**

Torque (kg-cm)	Voltage							
	5	6	7.4	11.1	14.8	18.5	22.2	
20:1	0.64	0.75	0.99	1.40	1.86	2.43	2.79	
35:1	1.10	1.32	1.63	2.44	3.26	4.07	4.68	
50:1	1.57	1.89	2.33	3.48	4.65	5.82	6.58	
63:1	1.96	2.38	2.90	4.40	5.86	7.33	8.79	
86:1	2.70	3.24	4.00	6.00	8.00	10.00	12.00	
115:1	3.61	4.34	5.35	8.02	10.70	13.37	16.05	
150:1	4.71	5.66	6.98	10.47	13.96	17.45	20.93	
250:1	7.88	9.43	11.63	17.45	23.28	29.08	34.89	
360:1	11.32	13.56	16.75	25.12	33.49	41.87	50.24	

Exceeding 8.84kg-cm (55oz-in) will damage the output gear stage. Do not stall the motors highlighted in red at the indicated voltage.

FIGURE 1



FIGURE 2



FIGURE 3

How Not to Die. I Mean, Build a Test Box

● by Matthew Spurk

You've just finished putting the finishing touches on your new

insect class combat robot. You've spent the last three hours tirelessly

wiring, soldering, grinding, tapping, cutting, and wrenching on your bot.



FIGURE 1. A storage trunk, a.k.a., the basis for a robot test box.

Now comes the most exciting time of the entire build process. That first time you flip on the power switch and bring your machine to life. It's a great time to be alive, isn't it? Wouldn't it be great to stay alive?

That first time you flip the switch is exciting, but it's also the most dangerous time in the build. Over the past several weeks, I've watched builder after builder posting videos on **YouTube.com** of their robot's first weapon test. A little robot, six pounds or less with aluminum, steel, or titanium spinning at 5,000, 10,000, even 30,000+ RPMs sitting on the floor with little or no protection between the bot and builder. I cringe and grit my teeth waiting for a loose screw or random nut (that was dropped on the floor and forgotten a mere two hours ago) to ricochet around the room. I further cringe as the video continues with the inevitable test hits. I'm a nervous wreck after 40 seconds of video.

Don't get me wrong. I've done

the same thing, and I've lived to tell about it. My whole perspective changed the day my vertical spinner "gyro'ed" over upside-down with the blade at top speed and struck the concrete driveway I was testing on. Pieces of concrete shot at me. I was fortunate I was not hit. That single event was a huge wake-up call. "HEY, 10,000 RPM IS REALLY FAST. HOW 'BOUT WE TRY NOT KILLING OURSELVES NEXT TIME, HUH?"

I'll give you a breakdown of how to build a small, simple test box that you can use to spin-up those nasty weapons, hit stuff, practice driving with the weapon on, and not DIE. Here's the best part: The whole thing can be built for under \$100 and in less time than it takes me to find a pair of matching socks in the dryer.

You will need the following items/tools for this project: a storage trunk (see **Figure 1**; they can usually be found in department stores for \$20 in late summer when college students are going back to school); a 14"x 28" piece of polycarbonate sheet 1/4" thick or greater; one 8' long 2"x 3" piece of lumber; a box of wood screws; some string or lightweight chain; a drill; a hole-saw or spade bit; and a jigsaw. If you have a particularly nasty robot, you can pick up a sheet of Medium-Density Fiberboard (MDF) for additional reinforcement (described later).

The process for building this box

is so simple it practically builds itself. (Okay, it's not that easy, but you could build a robot to build the box for you.)

Altering the Trunk

You have your trunk sitting in front of you. In theory, you could throw the bot in there, close and lock the lid, and call it a day. But unless you're a dolphin, it's hard to drive your robot via sound. The first thing you need to do is create an opening in the top of the trunk so you can see in. Driving a bot via looking at it is so easy, a caveman can do it. Measure in about 2" from each corner of the trunk; the dimension isn't super critical, so don't stress over it to much. Break out your trusty (or rusty) drill and spade bit/hole-saw and create a hole in each corner. The holes are for the purpose of starting the jigsaw. Next, break out the jigsaw and run a straight line between the holes forming a rectangular opening in the lid (see **Figure 2**). Now you could throw your robot in there and test, but the test box offers about as much protection as a tissue in a hurricane. (From experience, that isn't much protection at all).

Installing the Poly

What was once a perfectly good trunk is now a perfectly good trunk with a large hole in the top. It's time to install that

FIGURE 2. Create a rectangular opening in the trunk.



FIGURE 3. Polycarbonate sandwiched between the lid and the 2" x 3" piece of lumber.



FIGURE 4. Lightweight chain used to prevent the lid from opening completely.



polycarbonate. Open the lid of the trunk and set the polycarb into the opening. I like to turn the trunk upside down with the lid resting on the ground and the main box supported with scrap lumber for this part. Now take your 2" x 3" piece of lumber and cut it to fit the inside of the lid. The polycarb should be sandwiched between the 2 x 3s and the inside of the lid (see **Figure 3**).

Run your wood screws through the lid into the 2 x 3 around the sides. I prefer to not run the screws through the polycarb as I prefer to not put any holes in it. When you drill plastics, you create micro-cracks which over time and impacts can grow to become bigger cracks. If you're anything like me, you are very anti-crack. (Remember kids, cracks kill).

Test the Test

Functionally, the test box is now complete. You can throw your bot in there, drive it around, turn the weapon on, and everything will be hunky dorey. At this time though, you may notice something inherently annoying about your test box. By opening the lid to the complete open position, the test box will flop onto its back, and that's just not cool. There is a simple fix for this problem. Take some of that string or lightweight chain you have laying around and create a leash on the outside of the box that limits the opening of the

lid (see **Figure 4**). My test box allows motion to slightly beyond 90 degrees from the closed position. This allows the lid to stay open without my needing to hold it open, but it won't tip over either.

Upgrades

Bling Bling. Here are a few ideas that I've had that could really make your test box the crème de la crème, so to speak. Remember earlier when I told you to pick up that MDF? This is where it can come in handy. If you're worried about your bot getting through the plywood side walls, screw some MDF onto the outside of the test box. This will add weight, but for less than \$20 a sheet, you can't go wrong adding it.

Another upgrade you may want is new wheels — maybe even with spinners. Your test box, of course, will help you test your robots safely, but when you go to an event, you can use your test box to carry stuff. Bolt your test box to a two-wheel hand truck. Not only can you test your bot safely in the pits (get your frequency clip first), it will also save you trips from bringing your stuff in from the car. Nice. I didn't have a hand truck, but I did have an old skateboard I bought at a yard sale. It doesn't turn great, but I can ride my test box at the skate park.

You may also want to upgrade the floor of your test box with the



FIGURE 5. Completed test box with bot inside, ready to be tested.

same floor as the arena you'll be competing in. If the full size arena has seams, add a seam to your test box. Remember you're testing, so you want to know how your robot will react to the conditions it will likely face in combat.

Safety (Test) First

It has always surprised me how much people are willing to spend on a CNC titanium blade or custom turned bearing mounts for the bots, but those same spending habits don't translate into safety equipment. This simple test box will last you through many bots. The low cost of the materials and short build time will make it well worth the investment the first time your bot behaves abnormally or something catastrophic fails, and the high speed shrapnel is contained. Remember, the controls are much easier to operate if you have all your fingers. **SV**

C**o**mbat Zone's Greatest Hits

● by Kevin Berry

This month, CZ is experimenting with a new feature. We all know that creating bots is a fulfilling mental exercise, building them is satisfying craftsmanship, and going to events is great for bonding with your buds. But let's face it — it's about the

destruction! The audience doesn't cheer for box vs. wedge fests. It's the sparks, tosses, and large, flying chunks that get that "ohhhhh!" sound bouncing around the venue.

So, I asked a few builders to submit their own, personal "Greatest

Hits" photos. We'll continue this section as photos come in.

Before and after photos, a brief description of the fight, and the builder's name(s) can be submitted to me at **LegendaryRobotics@gmail.com**.

Extreme Robot Speed Control!

Sidewinder



\$399



- ◆ 14V - 50V - **Dual 80A H-bridges** - 150A+ Peak!
- ◆ Adjustable current limiting
- ◆ Temperature limiting
- ◆ Three R/C inputs - serial option
- ◆ Many mixing options - Flipped Bot Input
- ◆ Rugged extruded Aluminum case
- ◆ 4.25" x 3.23" x 1.1"

RC Control

BotsIQ Favorite

\$39.99



Scorpion Mini

- ◆ **2.5A** (6A pk) H-bridge
- ◆ 5V - 20V
- ◆ 1.6" x .625" x 0.25"

\$159.99



Scorpion XXL

- ◆ Dual **20A** H-bridge **45A Peak!**
- ◆ 5V - 28V
- ◆ 2.7" x 1.6" x 0.75"

\$104.99

2+ price

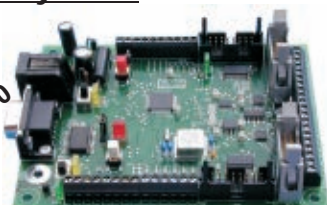
Scorpion XL

- ◆ Dual **13A** H-bridge
- ◆ 5V - 28V
- ◆ 2.7" x 1.6" x 0.5"

Dalf Motion Control System

- ◆ Closed-loop control of two motors
- ◆ Full PID position/velocity loop
- ◆ Trapezoidal path generator
- ◆ **Windows GUI for all features**
- ◆ Giant Servo Mode!
- ◆ C source for routines provided
- ◆ PIC18F6722 CPU

\$250



See www.embeddedelectronics.net

H-bridges: Use with Dalf or Stamp etc.

\$219



Magnum775

- ◆ RS-775 motor
- ◆ planetary gearbox
- ◆ 20:1 ratio - 700 rpm @ 14V
- ◆ Nearly 700W!
- ◆ Build something - rule BotsIQ!

\$89



\$79

Simple-H

- ◆ 6-28V **25A!**
- ◆ 2.25"x2.5"x0.5"
- ◆ 3 wire interface
- ◆ current & temp protection



www.robotpower.com

Phone: 253-843-2504 • sales@robotpower.com



Last Rites Hit.



Last Rites After.

Ray Billings submitted this month's item: Last Rites vs. VD6, at RoboGames 2009, in the 220 pound heavyweight class. Two of the biggest hitters at the event go weapon to weapon with spectacular results!

Dennis Beck submitted his Beetleweight, Utopia V2.0 for

its fight against Angry Dragan at Kilobots XVI, on January 17th, 2010. During the official match, Utopia's fight against Angry Dragan (a full body spinner) resulted in Angry Dragan getting thrown out of the arena before its shell could get spinning at more than 25% of its top speed. He proposed a grudge match, where Angry Dragan's shell would be allowed to get up to full speed before weapon-to-weapon contact would be made. The photo shows the resulting damage to Utopia from a

weapon-to-weapon hit. Props to Dennis for great sportsmanship!

Travis

Schmidt shows off his bot's damage. Mystery Box is a 6 lb Mantisweight. It went up against Chaos Theory's 14" carbide tipped sawblade at Saskatoon Combat Robotics Club's "Kilobots XIV" in September 2009. Plug request: www.kilobots.com. **SV**



Utopia.



Mystery Box Before.



Mystery Box After.

Photos courtesy of Dave Schumaker/ RockBandit, <http://rockbandit.net>.

ROBOGAMES



Compete at RoboGames!

Last year, over 1000 builders from around the world brought over 800 robots to San Francisco, in the 4th annual international event. This year, we expect even more robots and engineers to compete. Be one! With 80 different events, there's a competition for everyone - combat, androids, sumo, soccer, Lego, art, micromouse, BEAM, or Tetsujin! More than half the events are autonomous. Even if you just come to watch, you'll be overwhelmed with the diversity.

Last year, RoboGames hosted teams with over 800 robots from Argentina, Australia, Austria, Brazil, Canada, China, Colombia, Czech Republic, Denmark, Germany, India, Indonesia, Iran, Japan, Korea, Mexico, Netherlands, Peru, Singapore, Slovenia, Sweden, Taiwan, UK, and the USA.

Be a RoboGames Sponsor!

RoboGames is the world's largest open robot competition - letting people of any age, gender, nationality, or affiliation compete. Sponsoring RoboGames not only helps more people to compete, but also gets your company unrivaled press coverage and visibility. The event has been covered by CNN, ESPN, Fox, CBS, ABC, NBC (live), EBS Korea, NHK Japan, BBC, and countless print and web companies. Your logo can be everywhere the cameras turn!

Rent a Booth!

Booth spaces are at the front of the venue, ensuring lots of traffic. With 3000-5000 people each day, your company will get amazing traffic!



World's Largest Robot Competition
-Guinness Book of Records

North America's Top Ten Geek Fests
-Wired Magazine

SportCenter's Top Ten
-ESPN SportsCenter

"If you are a robot enthusiast, I would definitely encourage you to attend the RoboGames... Take a plane, train, space elevator, but definitely go!"

-Servo Magazine

"Impossible to Imagine, Impossible to Forget!"

-Robot Magazine

Events:

Combat: 340 lbs, 220, 120, 60, 30, 3, & 1 lbs

Androids: Wrestling, Demonstration, Stair Climbing, The Eagle, Door Opening, The Toss, Basketball, Lift and Carry, Marathon, Obstacle Run, Penalty Kick, Dash, 3:3 Soccer, Weight Lifting

Open Events: Fire-Fighting, Robomagellan, Maze/MicroMouse, Walker Challenge, Biped Race, Robot Triathlon, Line Slalom, Ribbon Climber, Vex Open, Lego Challenge, Lego Open, Aibo Performer, Balancer Race, Best of Show, Bot Hockey

Sumo: 3kg - Auto & R/C, 500g, 100g, 25g, Humanoid

Robot Soccer: Biped 3:3 & 5:5, Mirobot 5:5 & 11:11

Junior League: Lego Challenge, Lego Open, Lego Magellan, Woots & Snarks, Handy Board Ball, BotsketBall, 500 g Sumo, 120 lb combat, Best of Show, Vex Open

Tetsujin (ExoSkeleton): Lifting, Walking, Carrying

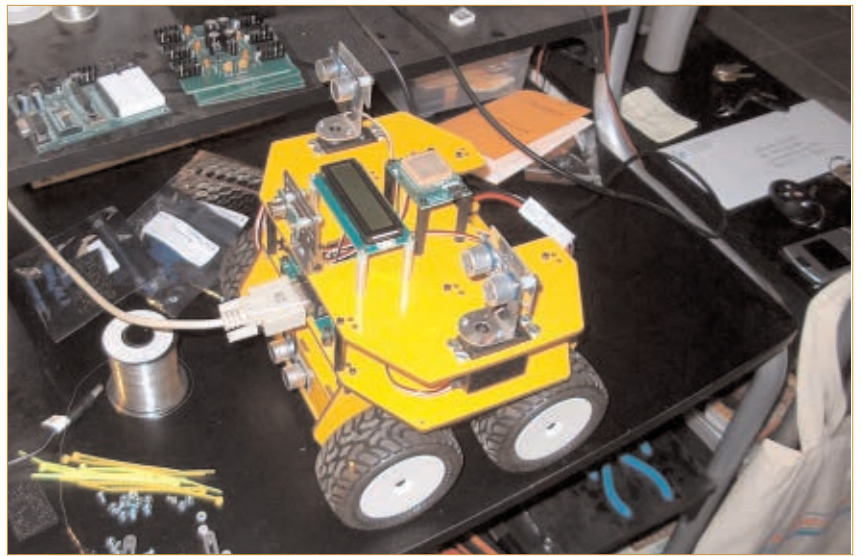
Art Bots: Static, Kinetic, Bartending, Musical, Drawing

BEAM: Speeder, Photovore, RoboSapien Hacker

April 23-25, 2010

San Mateo, CA

<http://RoboGames.Net>



GPS Navigation Part 2

In this article, I have built a small robot to test and prove the theories discussed in Part 1 from last month. Something we couldn't account for previously were the specifics of the robot since it hadn't been built yet. We also wanted to be able to use this concept and even the code elements on just about any platform.

The first thing to do in this case is to define several key elements of the design before we begin. How large is the robot going to be? How fast will it move? What type of terrain will it traverse? How will it steer?

For our purposes here, my answers were: about the size of a shoe box; no faster than three feet per second; asphalt and/or concrete; and skid steering. An old Lexan chassis I had lying around was perfect for this task (see **Figure 1**). My robot has four spur gear motors and a two-level platform for mounting the microcontroller and sensors.

Additional Components

I opted to use a Parallax BASIC Stamp 2p module for this system because it is more than capable of reading the raw NMEA data from the GPS receiver, parsing the data, reading the compass, and calculating headings from that data. It can then control the drive system to steer the robot where it needs to go, plus it simplifies the code design for those using a different platform for their robot and controller. The BS2p will mount on a Super Carrier Board which will provide connections to the remaining hardware. The chosen compass module for this application is the HM55B due to its easy integration and existing compatible source code. A serial LCD display, some PING))) sensors, and a heat array sensor (Devantech TPA81) have been installed on this robot. However, for this example, they will not be used and are optional for this part of the project. (These items will be used in Part 3. Be sure to check the Parallax website for the various components listed in this article.)

To drive the four gearhead motors, I chose two HB-25

FIGURE 1.
Old Lexan chassis from Lynxmotion.



FIGURE 2. HB-25 motor installation.

motor controllers due to their servo-like control and ability to not only handle these motors in pairs, but also to scale up should a larger robot chassis be used.

Construction

The first thing I did was assembled the chassis which was being repurposed from a previous robot project. The HB-25 motor controllers were installed in the belly of the chassis in between the motors. As you can see from **Figure 2**, it is somewhat of a tight fit. However, with proper cable routing, you can almost say it fits like a glove. The only thing I had to be cautious about was clearance for the fans on each HB-25. **Figure 3** shows the lower section of the robot assembled with one PING))) sensor installed and a power switch for the motors. Batteries were added, then the lower plate was installed along with some padding to hold the batteries in place (see **Figure 4**).

Two batteries are used in this design. One is dedicated to the motors and is switched independently using the power switch shown in **Figure 3**. The other battery powers the Super Carrier Board which, in turn, powers the GPS and

LCD (which both consume a lot of power). The batteries are standard 7.2V R/C battery packs from a hobby store. They are rated at 2,600 mAh each.

A separate switch controls power to the Super Carrier Board. As you can see in **Figure 5**, several three-pin headers have been mounted for connecting the various sensors and motors. The lower pins provide a provision for a pushbutton which is used to set waypoints for the GPS navigation.

Figure 6 shows how the LCD and GPS mount to the top plate, as well as the additional PING))) sensors and the thermal array sensor. This plate is then installed on top of the lower section of the robot using standoffs (see **Figure 7**).

Programming

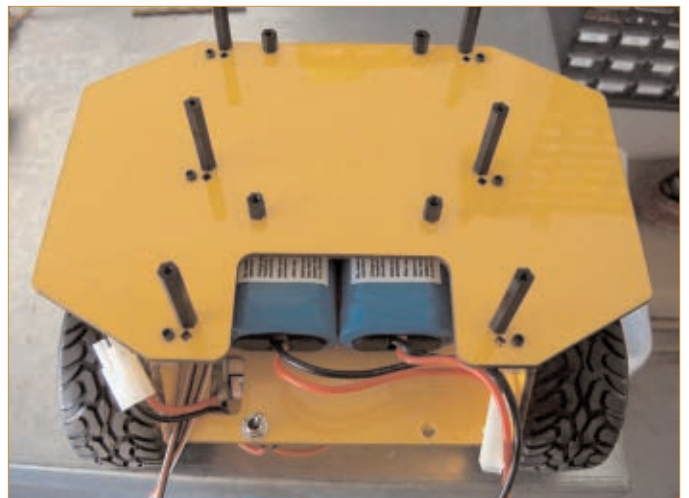
Once the robot construction was completed, it was time to download the program code (see **Figure 8**). The program has a main loop that does the following tasks until the last waypoint has been reached:

- Obtain current GPS coordinates.

FIGURE 3. Lower section assembly.



FIGURE 4. Batteries installed.



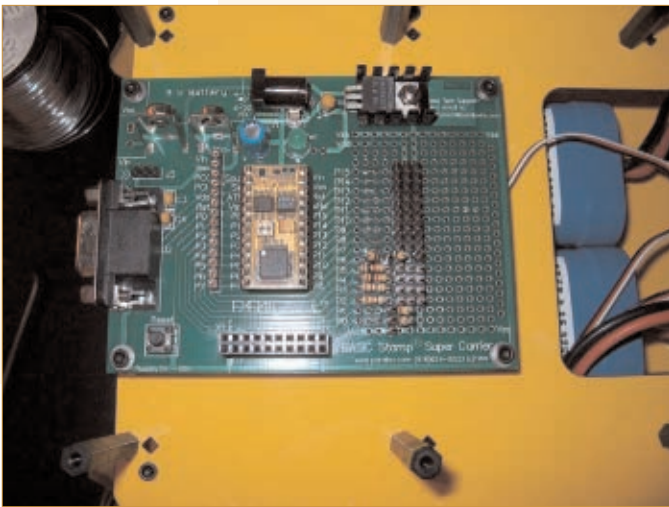


FIGURE 5. Super Carrier Board populated and installed.

- Calculate desired heading based on target position vs. current position.
- Obtain current heading from compass module.
- Calculate course correction based on desired heading vs. current heading.
- Determine if waypoint has been reached (and is last).
- Adjust course and speed of motor drive as needed to reach goal.

This loop will continue until the robot reaches its goal or the batteries die. As mentioned before, eventually other sensor data (such as from the sonar) will be factored into the equations to avoid obstacles, as well. For this article, let's assume it's a clear path (since mine was).

Code Details

The first task is to obtain the coordinates of the current position. These coordinates are used along with the target coordinates to calculate the desired heading. In Part 1, this was done using Pythagorean Theorem. You can also get

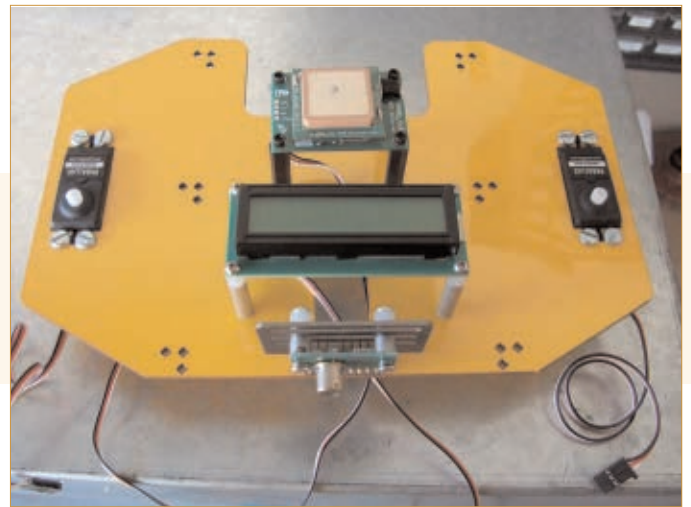


FIGURE 6. Top plate assembly.

the distance to target in this section using one more calculation. Next, we want to obtain the current heading from the compass module and compare it to our desired heading; any course corrections are then applied. At this point, the specifics of the motor drive system determine how best to handle the minor and major adjustments in course correction. On this robot, the skid steering makes any angle corrections easy since the robot can easily turn in place.

Taking the Robot for a Stroll

The initial results were quite promising from the first few runs; you can see video footage of the robot running the program by visiting the project page listed in the sidebar or on the *SERVO* website at www.servo-magazine.com. The only real issues were related to the weather which did somewhat impair GPS reception at a few points. If and when the GPS signal becomes invalid, the robot is programmed to stop in place and await a valid fix.

FIGURE 7. Top plate installed.

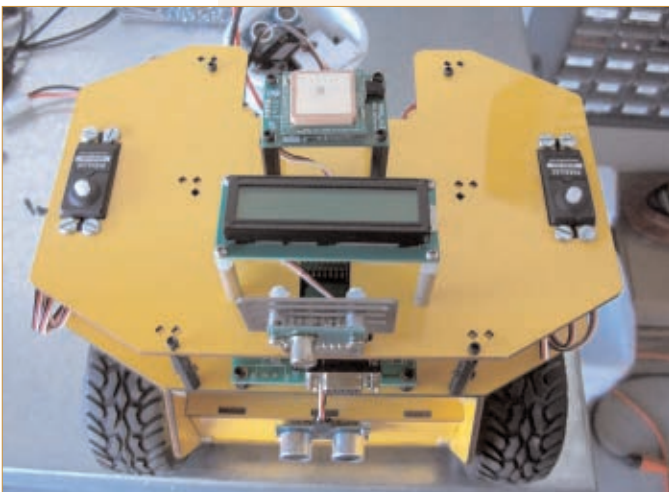
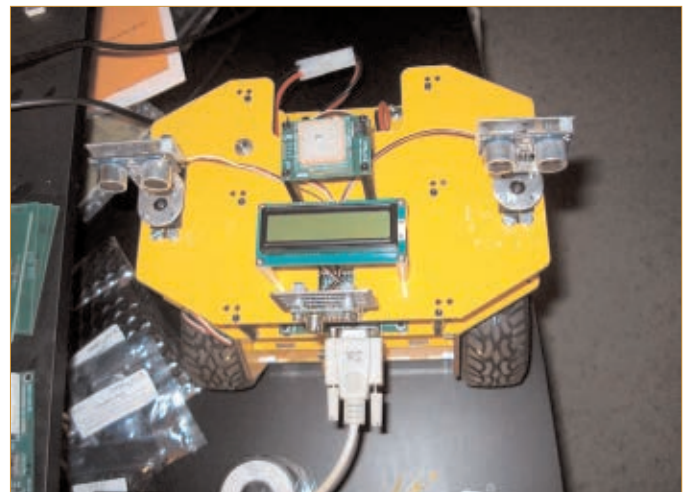


FIGURE 8. Ready for programming.



Resources

Parallax, Inc.
www.parallax.com

Project Page

www.savagelcircuits.com/gpsnav/

These types of behaviors can be adjusted via minor code changes I will cover in the next part. For the test phase (and lacking the sensors), I thought stopping was the safer way to go. Also, the corrections are somewhat jittery. Fortunately with the skid steering, it is not very noticeable. (The previous platform I tested the code on showed this effect much more.)

This will require some filtering in the code. The LCD and pushbutton functions are very rudimentary, but we'll be enhancing these, as well. Right now, the pushbutton serves only one purpose and the LCD only displays the current Longitude and Latitude. A larger display could show more, but a menu system allowing you to select different display modes would be easy enough to implement.

Final Thoughts

The size of the robot can be a help or a hindrance depending on your terrain. Remember, when testing GPS code like this fast is not necessarily better. Wait until your code is adjusted and fine-tuned before throwing the throttle wide open. I was very conservative with my speed settings while still allowing for a reasonable speed to travel the distances within the parking lot where my waypoints were located. Obviously, since GPS requires outdoor reception you won't be using a BoE-Bot for this purpose. On the other hand, something as large as a lawn mower might pose a threat to people or property if something should fail.

The robot chassis I chose was a great compromise of speed versus size, and has sufficient computing power to handle all the functions (including those not yet implemented). The code is well documented and even describes the connections to the motors. (Should you have any questions, I can be reached via the project site, as well.) Until next time, take care and have some fun!

SV



Enter the world of PICs & Programming with this great combo!

Getting Started with PIC's
Getting Started with PIC's
Getting Started with PIC's

Combo Price \$175.95

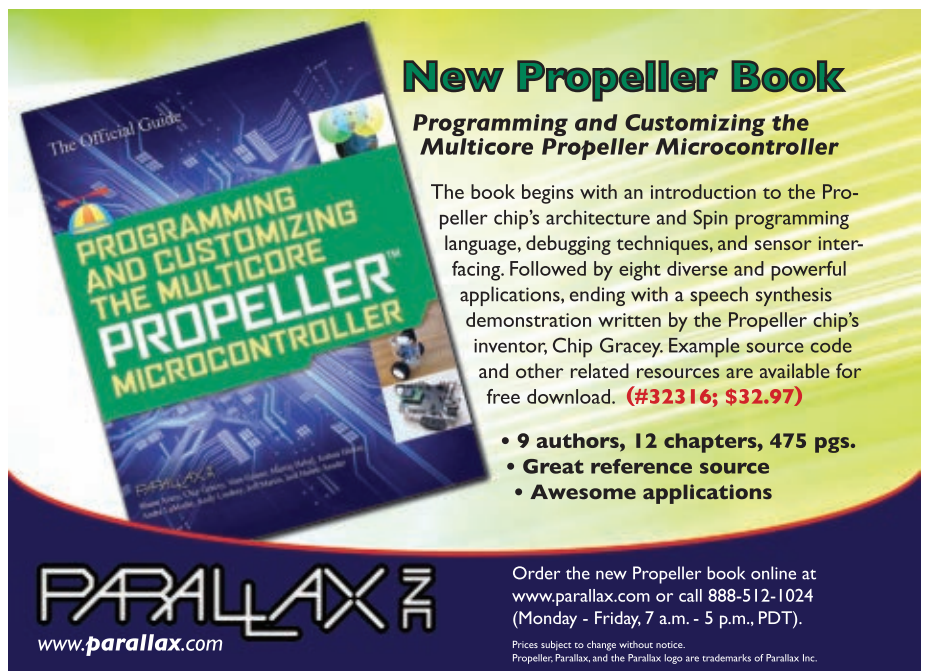
For complete details visit our webstore @ www.nutsvolts.com



MOTORS GEARBOXES WHEELS AND MORE

BB BaneBots

BANEBOTS.COM 970-461-8880



New Propeller Book

Programming and Customizing the Multicore Propeller Microcontroller

The book begins with an introduction to the Propeller chip's architecture and Spin programming language, debugging techniques, and sensor interfacing. Followed by eight diverse and powerful applications, ending with a speech synthesis demonstration written by the Propeller chip's inventor, Chip Gracey. Example source code and other related resources are available for free download. **(#32316; \$32.97)**

- 9 authors, 12 chapters, 475 pgs.
- Great reference source
- Awesome applications

PARALLAX

www.parallax.com

Order the new Propeller book online at www.parallax.com or call 888-512-1024 (Monday - Friday, 7 a.m. - 5 p.m., PDT).

Prices subject to change without notice. Propeller, Parallax, and the Parallax logo are trademarks of Parallax Inc.

The Bioloid Premium Kit Wrap-Up

By Rob Farrell

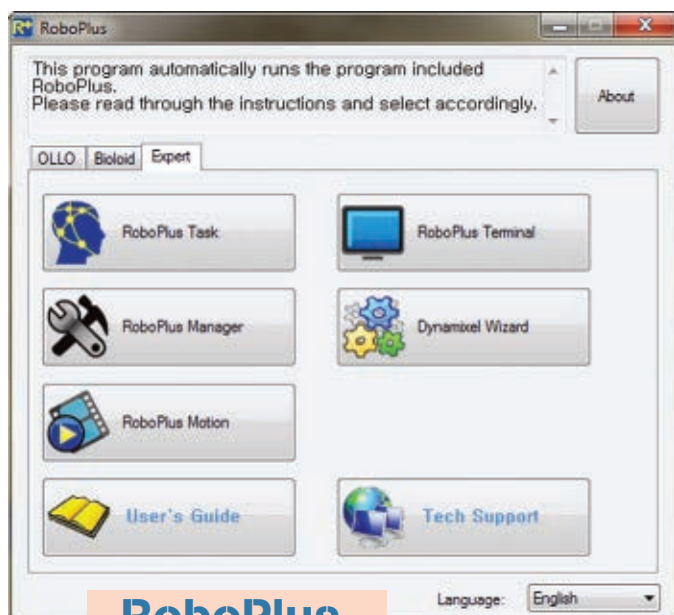


In the February '10 issue of *SERVO*, I covered some of the exciting new features of the hardware components and the assembly of the Bioloid Premium. This part of my review may feel a bit like a tutorial. However, my intent is to touch upon some of the vast capabilities of the RoboPlus software, point out some of its key features, and perhaps reveal a bit of why it's so impressive.

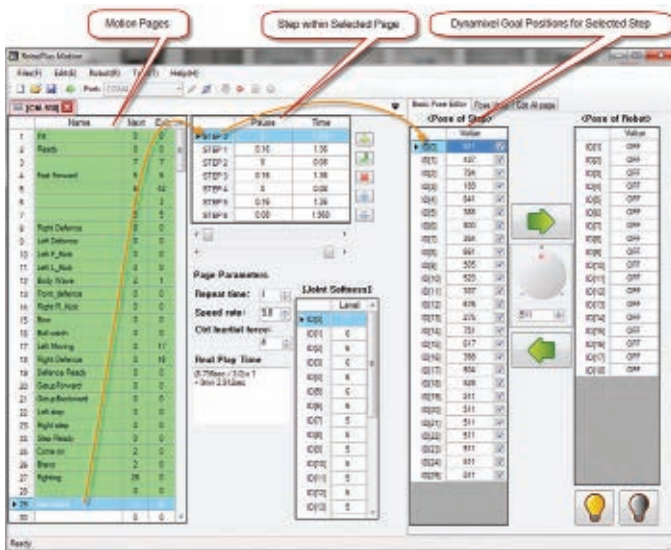
With the new Premium kit, ROBOTIS has released RoboPlus — a suite of PC software that includes RoboPlus Task, RoboPlus Manager, RoboPlus Motion, RoboPlus Terminal, and the Dynamixel Wizard. RoboPlus itself is a launch pad for these major applications, the user guide, and a link for ROBOTIS' latest online technical support. The latest version of RoboPlus supports both OLLO and Bioloid, and it includes a third tab for "Expert." The expert tab adds the Dynamixel Wizard and RoboPlus Terminal.

RoboPlus Motion is the new motion editor. With the motion editor, steps or poses are assembled into pages and pages can be linked together and repeated to form complex motions. Each page can contain up to seven steps and up to 31 Dynamixels. When the motion editor connects to the CM-510, it loads the directory of all the motion pages contained in Flash memory. Robot motion files can also be opened from a disk, in fact, several motion files and the contents of the CM-510 controller can be opened as tabs at the same time. For each tab or robot file, the pages are indexed in the left column followed by the page name, and next, the exit numbers. If next is nonzero, upon completion of the page the linked page will be played. Exit refers to the page that will execute when the robot receives an input from the RC-100 to execute a new motion. You'll note the CM-510 can contain 256 motion pages where the CM-5 can only contain 128. Pages that contain data are green while empty pages show as white. Clicking on a page places the steps of that page into the next set of columns. Each step has a "Pause" and "Time" setting associated with it. Pause is a delay between steps and Time is the time to complete the step. Each can be adjusted by clicking to select a step and then using the sliders below to adjust the values. Right below the steps are the page parameters such as the number of times the page should repeat, its overall speed, a calculation for the total page time, and the acceleration rate labeled as "Ctrl Inertial force." For each page, there is a setting for the joint softness or proportional gain for each Dynamixel.

The tables in the motion editor have a familiar spreadsheet feel. You can right-click on rows and copy or



RoboPlus.

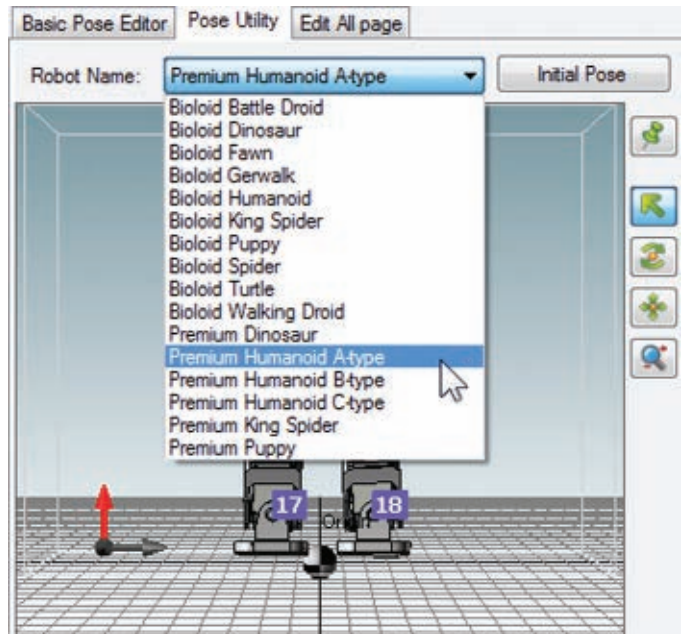


Motion Editor Markup.

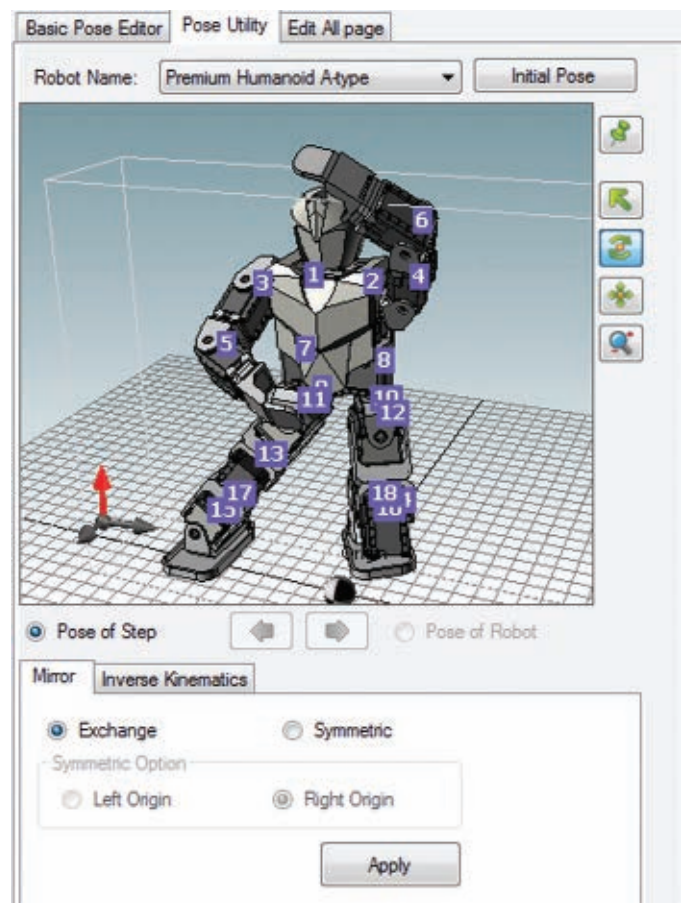
paste. This works for pages, poses, and individual Dynamixel settings. I found this very handy to copy a page from another robot file, or a step or pose from another page.

On the right-hand side of the interface are three tabs. The first is the "Basic Pose Editor" which allows adjustment of each Dynamixel. A single click instructs your connected robot to move to the selected pose. There are two small light bulbs at the bottom of the page. Click the dark bulb to disable torque or the yellow one to enable torque. While the torque is disabled, those Dynamixels can be repositioned by hand and re-enabled when in position. When you click on the light bulb to re-enable, "Pose of Robot" gets filled in with the current location of the attached Dynamixels. This is commonly referred to as "catch and play." The second tab is "Pose Utility" which opens up a 3D graphical view that can be configured from a large selection of standard robots.

If you build one of the standard configurations with this tool, you click on a page to select it and then click on a step; its pose is reflected in the virtual representation. The small icons to the right provide tips as you hover your cursor over them. Using them, you can center the view, select an object, rotate your point of view, zoom, and pan. Hovering over the icons displays tips: mouse the middle button to rotate the point of view, use the middle mouse button plus CTRL to pan, and use the scroll wheel for zoom. This makes it pretty easy to change your view to focus on one area or look from a different side at what your pose is doing. One of the really neat aspects of this is that a robot file can be loaded while the robot is not necessarily connected, and you can select a page such as the built-in walk. You can click the play icon at the top of the page and observe the page or a set of linked pages,



RoboPlus Motion Pose Utility.



Pose Utility Mirror.



and play it as it does normally on the robot in the virtual window. The interface lights up each step and page along the way. I found this quite instructive decomposing how the default walking gait worked. Note the diagram of the Bioloid Premium walking machine. It shows some of the complexity of walking and being able to change the direction while walking. The forward walk alone is comprised of four pages each with seven poses. Overall, there are 16 walking modes.

Beyond the 3D virtual representation, the pose utility has a couple other important features: mirror and inverse kinematics.

On the mirror tab, you can mirror a pose bilaterally. There are two methods: exchange and symmetric.

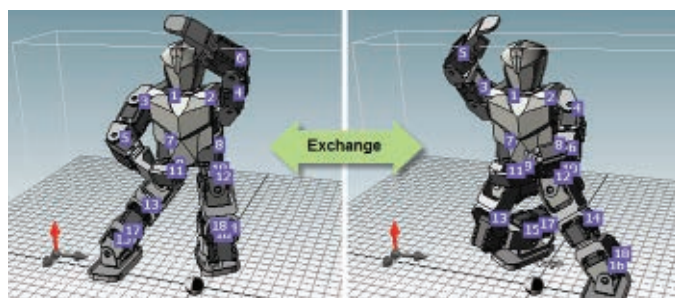
Exchange is what I would consider the classic mirror as shown in the figures here. Symmetric (as the name

implies) sets the robot's pose to be symmetric to either the left or right side.

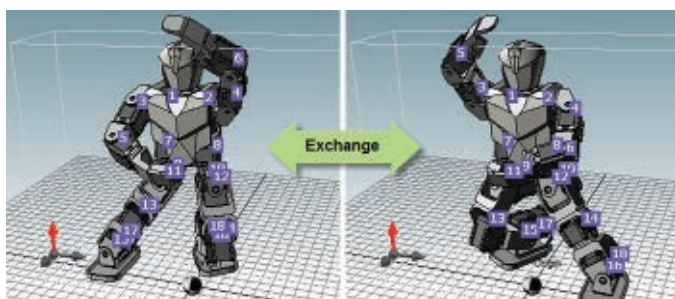
Due to the computational complexity, until recently inverse kinematics (IK) has largely been left to researchers. Now it's included as part of RoboPlus Motion — at least for the stock models. Inverse kinematics allows adjusting the position of the end effectors — in this case, a foot or feet — in Cartesian coordinate space and then computing the joint angles to achieve the goal Cartesian position.

ROBOTIS has included IK for both single foot and a walking step profile. The walking step uses both feet as end effectors and changes the use of the angular controls. In the single foot mode, the angle of the foot is changed with an angle change of ϕ , q , or y .

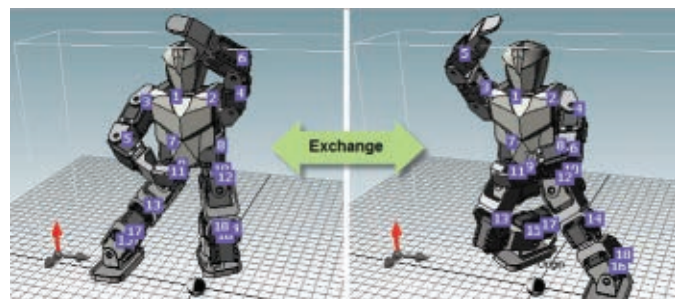
In "Walking Step" mode, both feet are end effectors and the X, Y, and Z adjustments are obvious; ϕ , q , and y



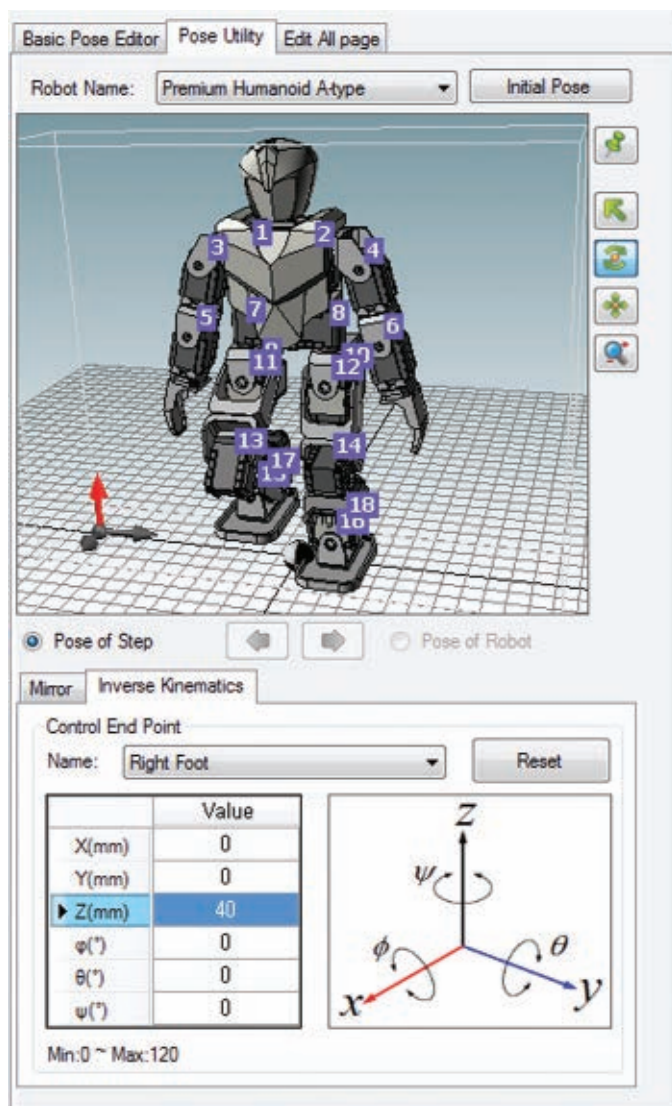
Pose Utility Exchange.



Pose Utility Symmetric Left Origin.



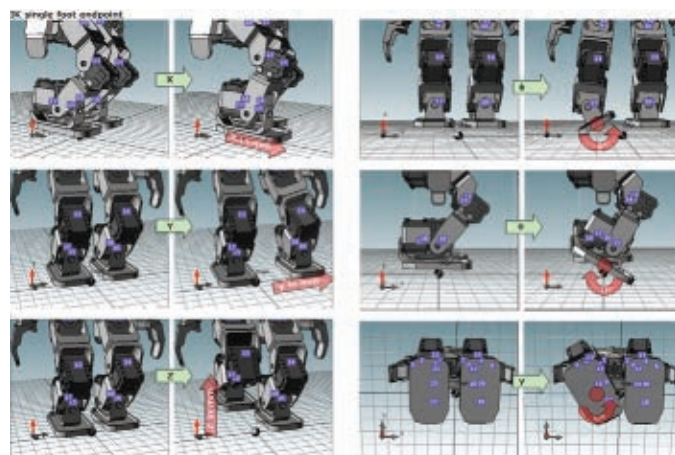
Pose Utility Symmetric Right Origin.



Pose Utility IK.



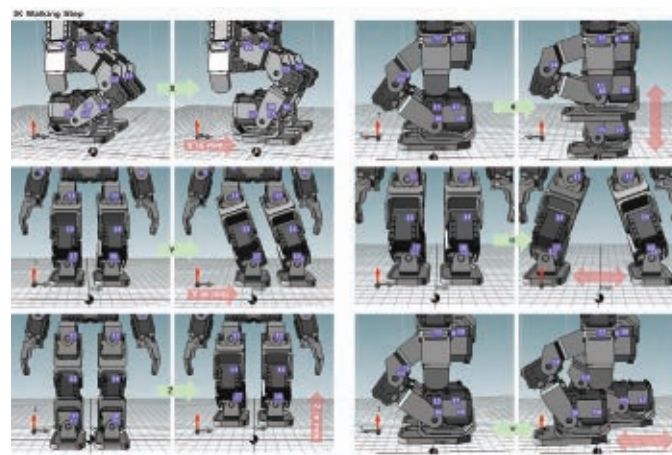
have been re-assigned to symmetrically adjust the feet, up or down, side to side, and back to front to help the user develop walking poses. I found it a bit odd that they no longer seem to be angles, but it does seem useful the way it's been designed. Overall, the motion editor appears a bit complex and cumbersome at first, but with a little practice becomes quite easy to use.



Inverse Kinematics Single Foot Endpoint.

Historically, experienced roboticists would use the Robot Terminal to quickly edit motion sequences. There are a few things in the new motion editor that can't be done with the Robot Terminal interface: cut, paste, copy, robot file management, mirror, and inverse kinematics.

RoboPlus Task is the programming interface that allows you to build behaviors by stitching together inputs



Inverse Kinematics Walking Step.



Task.

- Robot Premium Humanned - RC default configuration**
- Walking : U / L / D / R
 - Change Posture : 1 + U / L / D / R
 - Demonstration Moves : 2 + U / L / D / R
 - Soccer Moves : 3 + U / L / D / R
 - Battle Moves : 4 + U / L / D / R



Walking (U / L / D / R)			
U	Forward	D	Backward
L	Turn Left	R	Turn Right
U + L	Walk Forward + Left	D + L	Walk Left Sideways
U + R	Walk Forward + Right	D + R	Walk Right Sideways

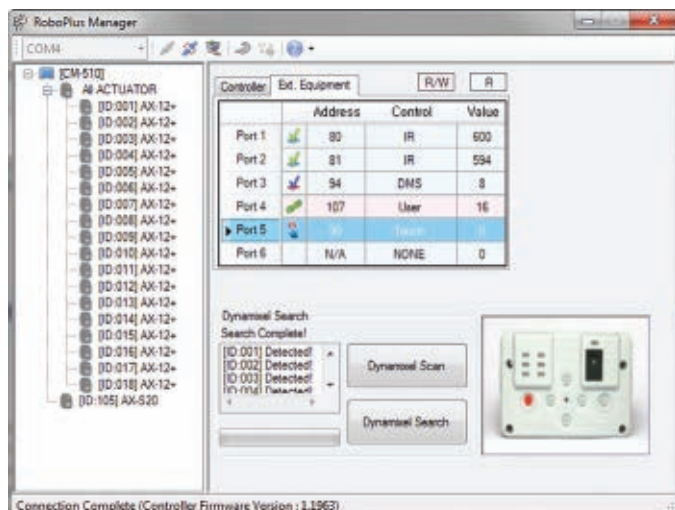
Posture 1 (1 + U / L / D / R)			
1 + U	Getting up Backward (When lying on stomach)	1 + D	Getting up Forward (When lying on back)
1 + L	Push-up	1 + R	Handstand

Posture 2 (2 + U / L / D / R)			
2 + U	Pound Chest	2 + D	Scratch Head
2 + L	Cheer	2 + R	Bow

Soccer Moves (3 + U / L / D / R)			
3 + U	Block Right (Release button to return to normal position)	3 + D	Block Left (Release button to return to normal position)
3 + L	Shoot with left foot	3 + R	Shoot with right foot

Battle Moves (4 + U / L / D / R)			
4 + U	Attack	4 + D	Defend (Release button to return to normal position)
4 + L	Attack Left	4 + R	Attack Right

Remote Control Modes RC-100 Default Task and Motions.



RobotPlus Manager.

and sensors, output motions and sounds, and (in some cases) a little math. The preinstalled default task makes for great reading when you get started. It provides a wealth of examples. The new interface is its own language and it has some similarities to C, but is a bit more graphical and simplified.

Only basic math operators are available and only one operator can be used in each line of code. This makes things a bit cumbersome to implement a complex calculation, but not impossible. Perhaps one of the most powerful features of Task is the callback function. The ROBOTIS motion engine updates servos every 7.8 ms as part of its normal interpolation. Each 7.8 ms (after updating the Dynamixel positions), the motion engine executes a callback routine. The callback can contain a small amount of code that can perform operations such as adjusting Dynamixel offsets using gyro or accelerometer input.

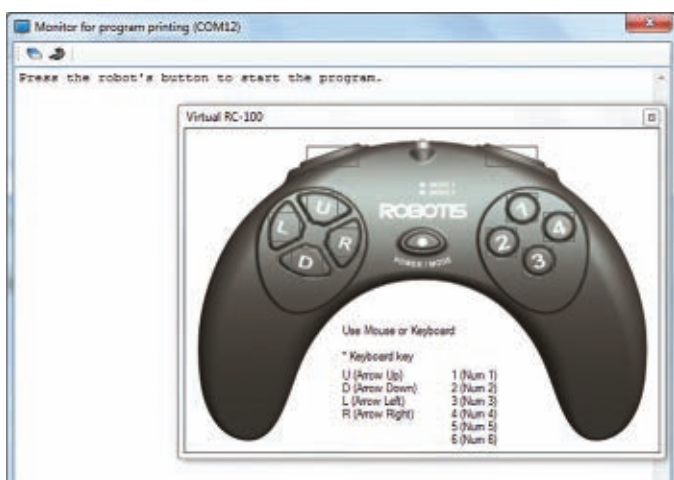
I found the search function a bit non-intuitive because it only searches for variables, not function names or other terms. I also found myself wishing I had a way to use a simple text editor to write my programs so I could easily search and print in a format I like. RoboPlus Task has a print function, but it leaves a lot of white space and uses large fonts, therefore creating lots of pages. It looks nice but being a long time programmer, I'm overly picky. In the end, these editing issues are only minor.

I found I was able to put together a simple PID algorithm using the gyro module and the callback function fairly easily. I have spent considerable time writing code for other robot basic interpreters but given my new experience with RoboPlus Task, I'd have to say I prefer it and it is considerably more capable than the others. It's a quick, solid tool for building some decent behaviors.

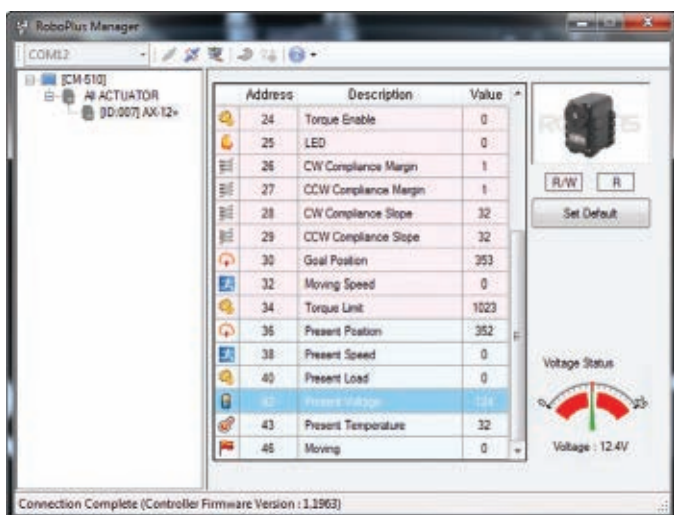
Under the "Program" dropdown menu, there is a selection for "view print of program" which is a serial terminal window that allows monitoring of print statements in the loaded program, but it also has a small RC-100 icon at the top. Clicking this icon reveals a virtual RC-100 controller that can be used while the program is running instead of the physical RC-100 which shares the same serial port on the CM-510. (Obviously, you can't monitor print statements and use the physical RC-100 simultaneously.)

The default Task included for the humanoid configurations is complete enough that one could stop at the assembly and use the robot successfully to play soccer or compete in Kung-Fu. I've always found it's easier to start with a good set of examples and ROBOTIS has really come through here. The default Task provides an interface to select from testing your assembly, to autonomous operation, to remote control operation.

RoboPlus Manager is a basic application that one only uses periodically. It's much like a robot device manager to



Print Window Virtual Remote.



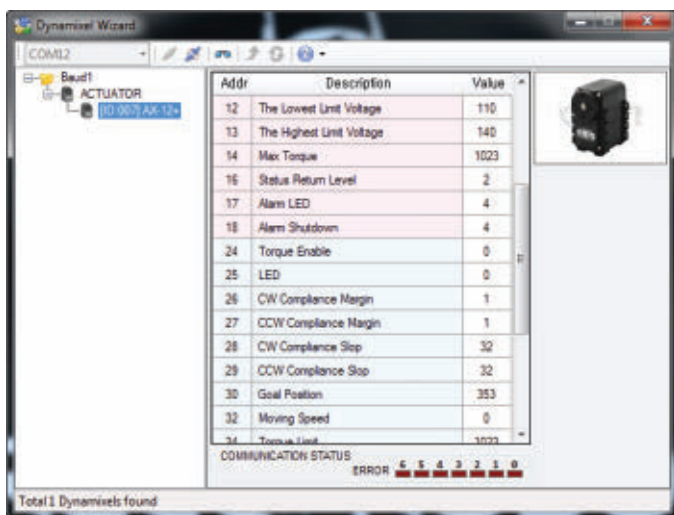
RobotPlus Manager 2.

do things such as reconfigure or test a Dynamixel, or to test a sensor through the CM-510 or other controller. This is an important tool to test and configure sensors, and to later test your own sensors. The RoboPlus Manager program also has the capability to update the firmware on the robot controller and if (for some reason) your firmware should ever become corrupt, the RoboPlus Manager has the ability to restore the firmware to an unresponsive controller. The manager also has the capability to configure ROBOTIS' optional Zig2Serial and ZIG-100 wireless interface for the PC so you can program and control your robot wirelessly from your computer.

The Dynamixel Wizard has similar capabilities for Dynamixels to the RoboPlus Manager, however, its interface is intended for use directly from a PC with the use of a USB2Dynamixel or similar serial to TTL interface (with no CM-510 controller serving as the mediator). It has a quick, convenient way to search for Dynamixels but is a little less graphical than the RoboPlus Manager.

RoboPlus Terminal is a basic terminal that has some added functionality to transfer data to and from the robot controller over a serial connection and only shows up on the advanced tab as it's intended for experienced users. The terminal is a text-based command line interface that allows keyboard input for motion editing, Dynamixel configuration, and bootloader access for firmware updates. Being a long time user of ROBOTIS products, for me, this represents access to the heart of the machine. From the Terminal, one can read and write directly to onboard Flash, RAM, and EEPROM. For those of us who want to build our own firmware, this is the stepping on point.

Using the terminal in program mode, you can edit motion pages directly using command line entries and cursor keys. Some details of motion editing from the

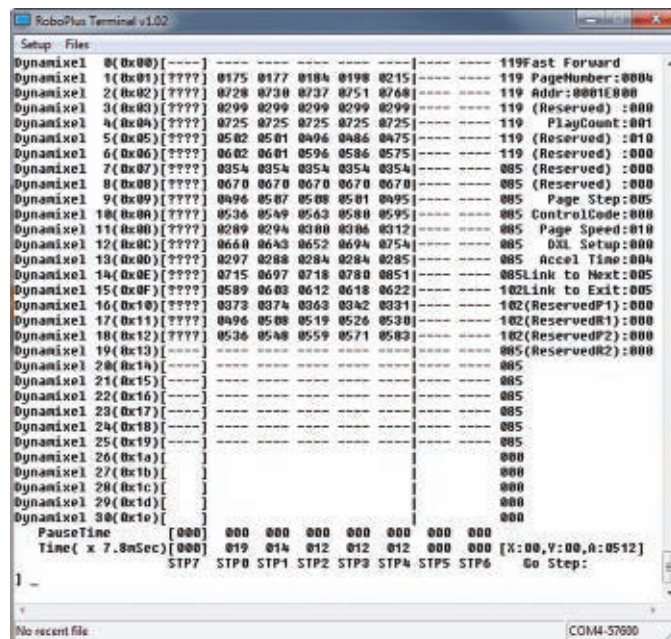


Dynamixel Wizard.

terminal mode are covered in the user's manual but to really understand the details, you have to search the net for prior documentation from the original Bioloid and the cycloid, and CM-2 and motion 512. In manage mode or monitor mode, you can use the terminal to read and write Dynamixel register contents and send commands to one or many Dynamixels. The terminal can also be used to monitor print statement outputs from task programs.

Conclusions

With RoboPlus, ROBOTIS has certainly put together an amazing suite of software that supports users of all skill levels. Often when I think about the value of things, it comes down to how much time I spend with them and how much enjoyment I get out of them. When I think about the fun and the experience that can be gained by any individual with the Bioloid Premium kit and RoboPlus, a smile comes to my face. If you have an interest in robotics and you're wondering where to start or, if you're an experienced roboticist investigating next opportunities or an educator or parent looking to inspire others, I can't think of a better place to start. **SV**



Terminal Motion Editing.

Look Who's Talking!

By Fred Eady

Explore this simple method that lets you and your robot play together in a wireless world!

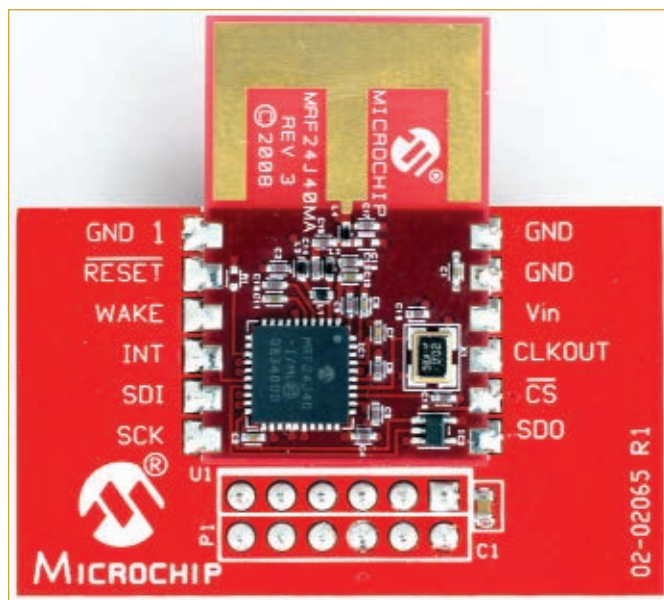


PHOTO 1. This is the MRF24J40MA module variant that is designed to interface to the PICDEM Z development platform. Note that the surface-mountable version of the MRF24J40MA module is actually soldered onto the PICDEM Z 12-pin socket carrier printed circuit board.

Robots are much like people. There are soldier robots and there are hunter-gathering robots. There are thinking robots and robots that perform physical work. Robots also have another commonality with their human counterparts. They need to communicate. This communication can occur in the form of internal system to system, robot to robot, human to robot, or robot to human. This month, we're going to explore a very simple wireless method of invoking internal-to-internal, machine-to-machine, and machine-to-human data communications using the bus that ZigBee rides — 802.15.4.

What is 802.15.4?

Formally known as IEEE 802.15.4, the 802.15.4 standard specifies the rules for the MAC and PHY layers used for low power networks. The MAC (Media Access Control) layer is responsible for logically handling the transmission and reception of data that passes through the PHY. The PHY layer is comprised of the actual hardware that moves the data. The hardware can be wire or a radio. The most popular protocol implementation of the 802.15.4 standard is ZigBee which uses 802.15.4 to fly ZigBee packets around in a ZigBee network. The low power networks targeted by 802.15.4 are commonly used for monitoring and control. ZigBee — which is designed for monitor and control purposes — is a good example of a typical 802.15.4-based network. The IEEE 802.15.4 implementation is actually a part of the 802.15.4 wireless PAN (Personal Area Network) specification. PANs are simple packet-based networks that usually support sensors and battery-operated devices.

The interesting aspect of battery-powered devices that work on PANs is their battery life which is commonly measured in years instead of hours. The factors that contribute to a PAN node's long battery life are the length of the messages and the distance between nodes. PANs more often than not are designed to transfer low

In the discussion that follows, we will examine and exercise the capabilities offered to us by the 802.15.4 specification. Instead of building up an application-specific piece of PAN hardware, we'll generate some 802.15.4 PAN traffic using an off-the-shelf development kit from Microchip. The idea is to show you how 802.15.4 works with a very basic PIC microcontroller implementation that you can use as a reference design to build your own PAN hardware.

The Microchip MRF24J40MA IEEE 802.15.4 2.4 GHz Transceiver Module

As its name implies, the MRF24J40MA 2.4 GHz transceiver module is an 802.15.4 compliant RF transceiver. In that the MRF24J40MA is based on 802.15.4, it can natively support ZigBee and the numerous proprietary 802.15.4-based wireless network protocols such as Microchip's MiWi and MiWi P2P. Like the majority of 802.15.4 radio equipment, the MRF24J40MA transceiver uses the unlicensed 2.4 GHz frequency band and has enough output power to reach out and touch other 802.15.4 nodes at up to 400 feet.

The MRF24J40MA transceiver module that we will be discussing is shown in **Photo 1**. Note that my module is specifically pinned to mount on a PICDEM Z development

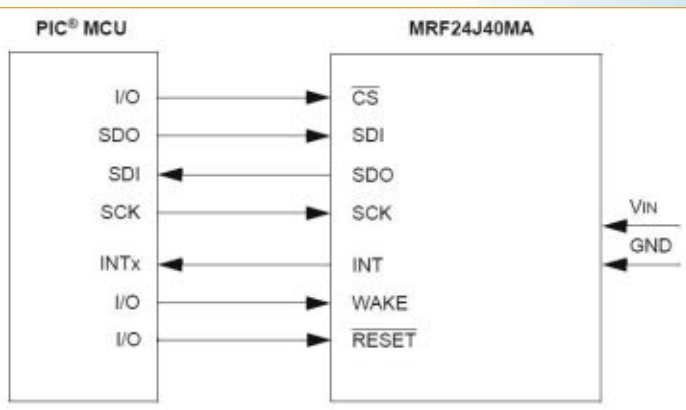
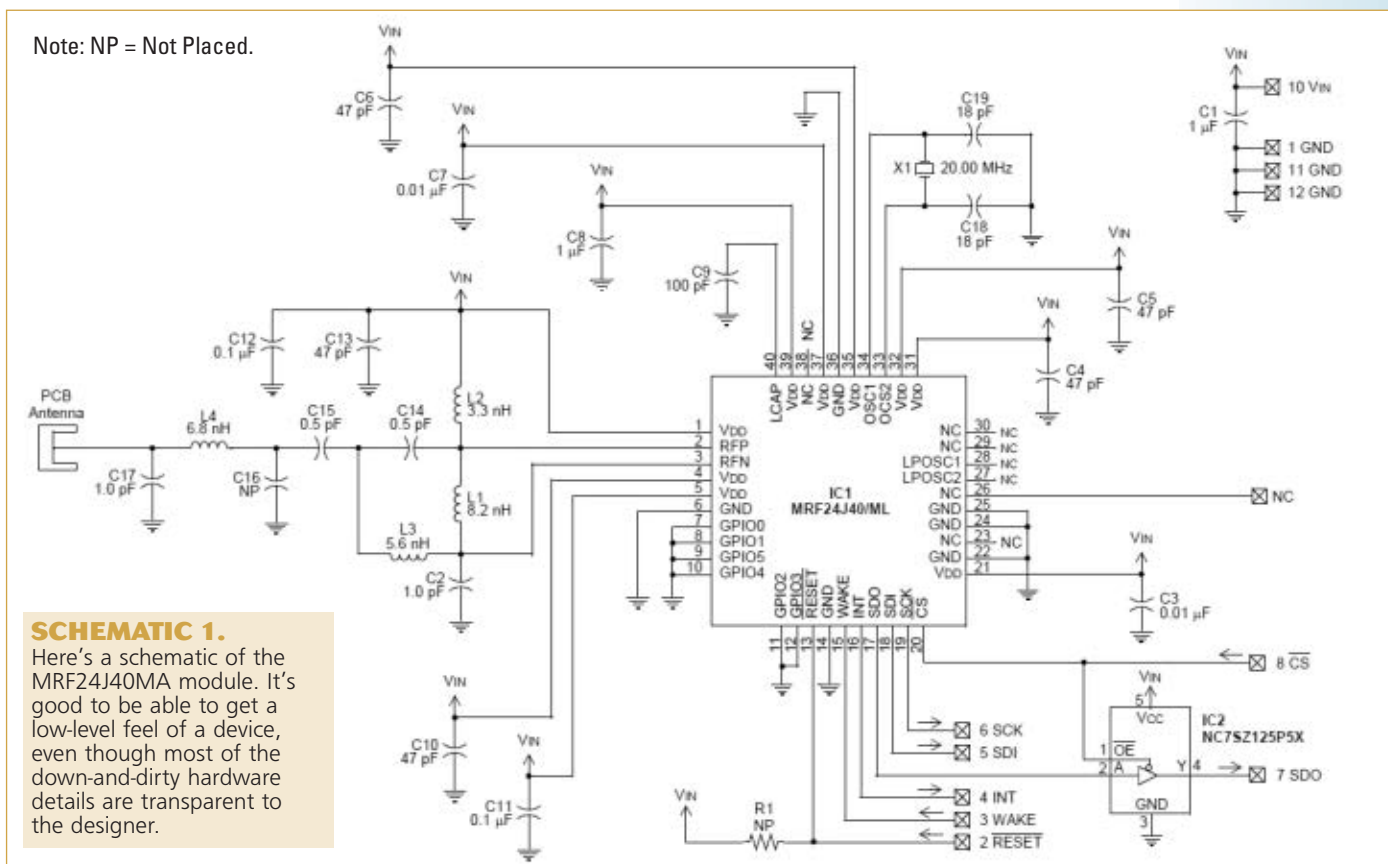


FIGURE 1. No rocket science here. This is a standard SPI hookup. In fact, if you recall our recent ZeroG discussion, this is almost identical to the SPI interconnect between a PIC and the ZeroG wireless Ethernet module.

board. The 12-pin connector includes electrical paths to supply +3.3 volts to power the MRF24J40MA, a four-wire SPI portal interface, a reset line, and wake and interrupt control lines to the host microcontroller. The design of the PICDEM Z connector defies you to incorrectly mount the radio onto its development board or any other host platform. If you would like to have your own PICDEM Z 12-pin connector assembly, you can get one from Samtec by specifying part number LST-106-07-F-D. The MRF24J40MA radio module can also be had in a 12-pin surface-mountable package, as well. If you look carefully at **Photo 1**, you'll see that the 12-pin SMT variant of the MRF24J40MA module is soldered onto the PICDEM Z 12-pin socket carrier printed circuit board (PCB).



There is no simpler way to do this than through its EUSART. If you focus just above the PIC18LF4620 in **Photo 2**, you'll see an assemblage of ceramic capacitors surrounding a MAX3221 RS-232 converter IC (U5) that just happens to be attached between the PIC's EUSART and the outside world.

Just to the right of the MAX3221 lies the 3.3 volt regulator area which is built around an LP2981 (U2). The LP2981 is rated for a maximum input voltage of 16 volts and is a good choice for this design as the PICDEM Z host board can power itself from a standard nine-volt battery or the Microchip PTF09051 power brick.

Photo 3 is a fly-over shot of the 12-pin MRF24J40MA host interface connector. As you can see in **Figure 2**, not all of the 12 available connections are put to use. An interesting point in **Figure 2** is the use of MISO and MOSI instead of SDO and SDI. Most Microchip documents will reference the SDO/SDI combination which (when translated) means Serial Data Out and Serial Data In, respectively. In **Figure 2**, MISO – which is short for Master In Slave Out – is synonymous to Microchip's SDI, while SDO is French for MOSI (Master Out Slave In). SPI portal connections work just like direct-linked RS-232 connections. The output of Node A must connect to the input of Node B, and vice versa. That is the Master node's MOSI signal must route to the Slave node's MISO, and the Slave's MOSI must feed the Master's MISO.

The SPI signals that are used to converse with the MRF24J40MA are shared by a Microchip TC77 thermal sensor (U3) which can be partially seen in the lower portion of **Photo 3**. The MRF24J40MA is selected via pin 6 of the 12-pin PICDEM Z host connector. I/O pin RA2 of the PIC18LF4620 is used to select the TC77. Only one SPI slave device can be selected at any time.

A full view of the PICDEM Z host board is under the lens in **Photo 4**. A couple of user-accessible LEDs and pushbuttons are present to allow the designer to simulate inputs and view output status. That's the drill on any development board.

Without going into the PIC18LF4620 circuit details, we now know that the MRF24J40MA moves data between itself and the PIC18LF4620 using a standard four-wire SPI portal. We also know that there is a TC77 thermal sensor that shares the four-wire SPI connection. If we require them, a standard RS-232 port, a set of pushbuttons, and a pair of LEDs are available to us. I purposely didn't post a PICDEM Z host board schematic as you can get a complete one by downloading the PICDEM Z Demonstration Kit User Guide from the Microchip website.

Let's Mix It Up

Okay. We have the lowdown on the MRF24J40MA. We also have a very good idea about how the PICDEM Z host is put together as it is based on a general-purpose PIC18LF4620. It's time to meld the radios to the host boards and transfer some data.

The reason we're here is to gather enough information to enable us to use a microcontroller like the PIC18LF4620 to put some MRF24J40MA 802.15.4 radios on the air. So,

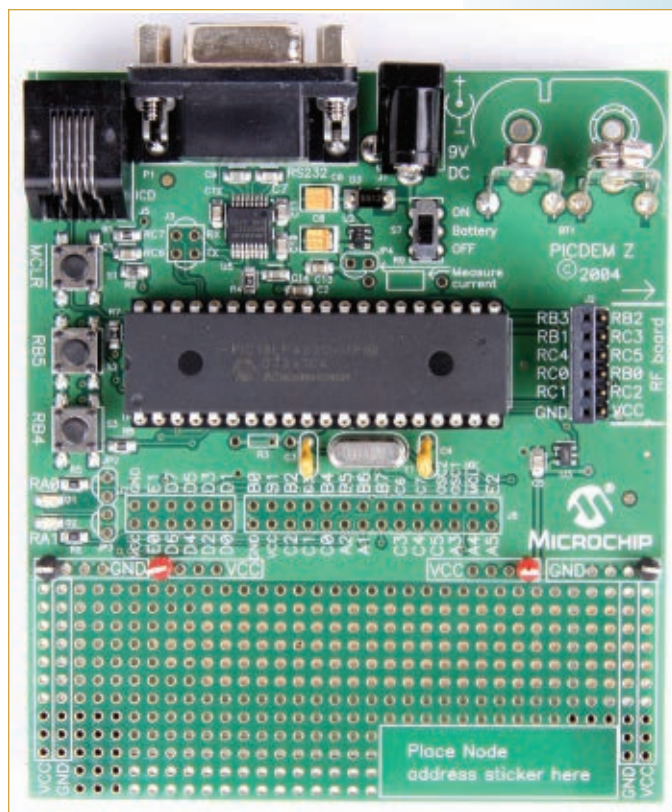


PHOTO 4. This is a view of the complete PICDEM Z host board. The normal complement of pushbuttons and LEDs make up the rest of this development platform.

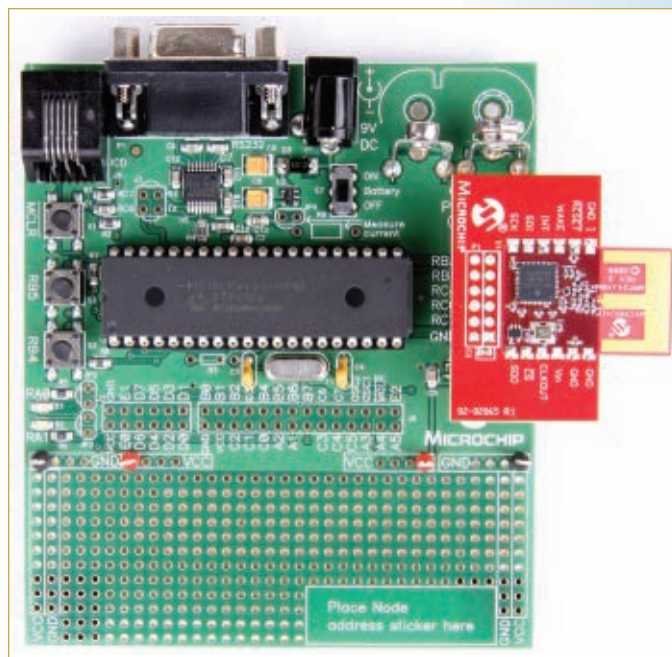
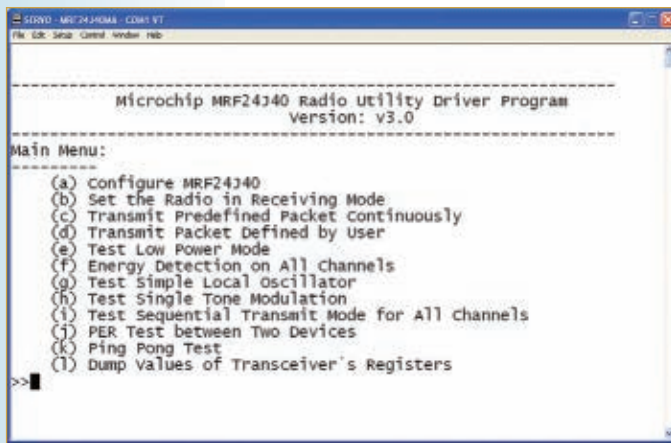
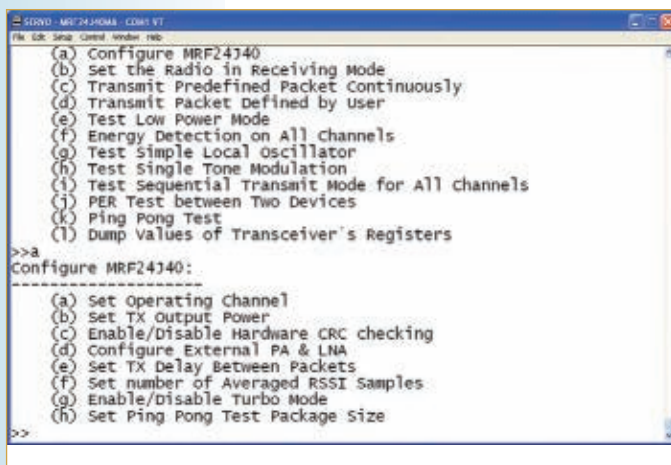


PHOTO 5. The PICDEM Z host/MRF24J40MA module combination you see here is waiting for some radio driver firmware. Actually, there is a pair of these waiting to be programmed. One host will eventually be programmed as a PAN Coordinator and the other as an End Device.

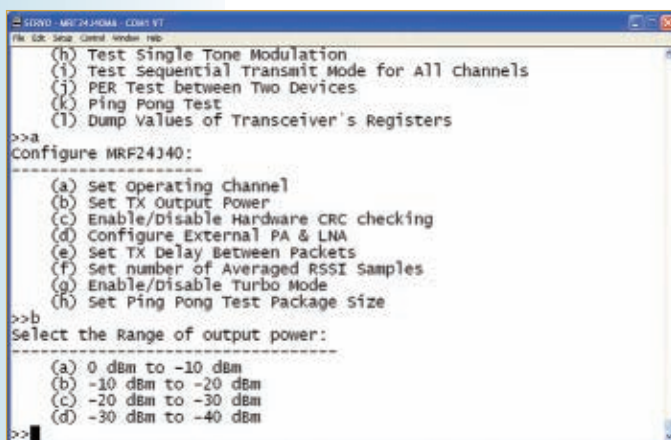
the first order of business is to mate the PICDEM Z host board and the MRF24J40MA module which I've done in **Photo 5**. To be able to form a network, I created another MRF24J40MA node by loading a second PICDEM Z host



SCREENSHOT 1. This Tera Term Pro capture lists everything we can do with the MRF24J40 Radio Utility Driver. It looks like this could be fun even without having to put on a pointy star-studded hat.



SCREENSHOT 2. There are some really interesting menu options here, as well. However, our goal is to send and receive some data. So, I figure we won't be configuring any external PAs or LNAs today.



SCREENSHOT 3. As you can see in this capture, the lowest power output we can dial in is between 0.001 and 0.0001 milliwatts. The MRF24J40MA is able to track a signal with a single strength as low as -94 dBm. The MRF24J40MA's receiver overloads at anything above +5 dBm.

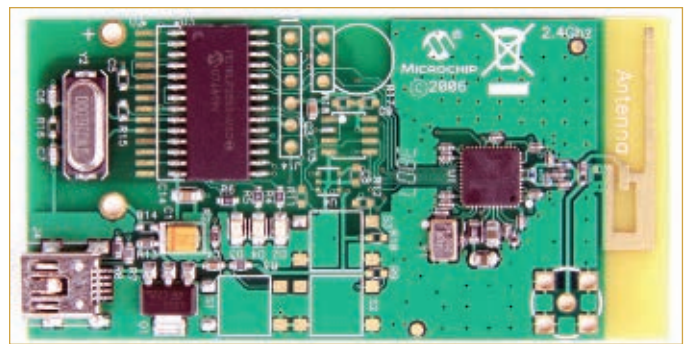


PHOTO 6. If you really want to get serious with MiWi or ZigBee, you need one of these.

board with a second module.

I've downloaded the MRF24J40 radio utility driver which we'll use to initially expose the capabilities of the module. The radio utility driver will allow us to configure the MRF24J40MA and run basic transmission and reception tests. The radio utility driver package comes as a collection of MPLAB projects with precompiled hex files. So, we only need to program the PICDEM Z hex file into each of the PICDEM Z/MRF24J40MA board pairs. After I set up a Tera Term session for 19200 bps, eight data bits, no parity, and one Stop bit for each node, we should be ready to rock and roll. According to the utility driver user information, we should be welcomed by a Main Menu by way of the PIC18LF4620's RS-232 port and Tera Term Pro. Behold **Screenshot 1**.

What moves you in **Screenshot 1**? Personally, I really want to jump to the Ping Pong Test, but I think we oughta start by punching in "a." Okay, "a" it is. Here we go again. What moves you in **Screenshot 2**? I like "b." Recall that the MRF24J40MA datasheet specified a typical range of 400 feet. Well, now we know how far 1.0 milliwatts (+0 dBm) off transmit power will take us. I chose to keep the output power at +0 dBm.

I messed around with the other MRF24J40 radio utility driver menu options and finally broke down and ran the Ping Pong Test. As you can see in **Screenshot 4**, there's nothing new to report. The radio utility driver is definitely aimed at the RF engineers that wear those funny hats. Let's go to Las Vegas.

Elvis Did It MiWi

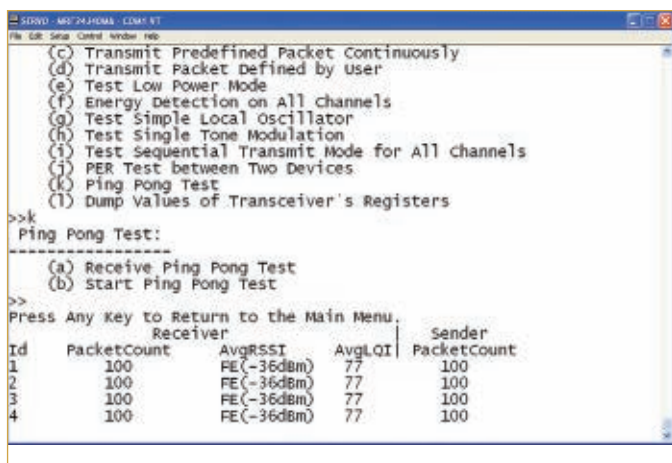
MiWi is a Microchip protocol that can be used in very simple wireless 802.15.4 networks. Our MiWi network will consist of a MiWi Coordinator and a MiWi End Device with both nodes configured as shown in **Photo 5**. The MiWi package is a free download from the Microchip website and comes with ready-to-run code for a PAN Coordinator and End Device. I've loaded up the PICDEM Z hosts. So, let's get the PAN hot.

Here's the plan. I'll start up the MiWi Coordinator first. It will search for a pre-existing network. Naturally, it won't find one and it will start a network of its own. Next, I'll fire

up the End Device. The End Device will attempt to join the PAN Coordinator's new network. Once the End Device joins the newly created PAN, I'll press the RB4 button on the End Device which will send a byte of data to the MiWi Coordinator. The MiWi Coordinator receives the report (French for PAN packet) and toggles the LED controlled by I/O pin RA1. Again, I will press the RB4 button on the MiWi End Device. Another report will be transmitted which will again toggle the LED on the MiWi Coordinator host board. While all of this is going on, I'll be capturing the packet flow using a ZENA MiWi Sniffer like the one you see in **Photo 6**.

The results of the plan as reported by the MiWi nodes are listed in **Screenshot 5**. Let's associate the ASCII MiWi messages with the ZENA data captures beginning with **Screenshot 6**. As the text in **Screenshot 5** states, the MiWi Coordinator is looking for a pre-existing network to join. After transmitting a number of Beacon Request frames, the MiWi Coordinator gives up and plants a stake in the ground creating its own network.

Immediately after power-up, the MiWi End Device begins its search for a suitable network to join. The only PAN in range is the newly created 0x011B PAN which is the MiWi Coordinator's domain. The MiWi End Device tells us about the network and node it found in **Screenshot 5**. The ZENA capture you see in **Screenshot 7** reveals that after discovering the new PAN, the MiWi End Device sent an Association Request to the PAN's MiWi Coordinator. A



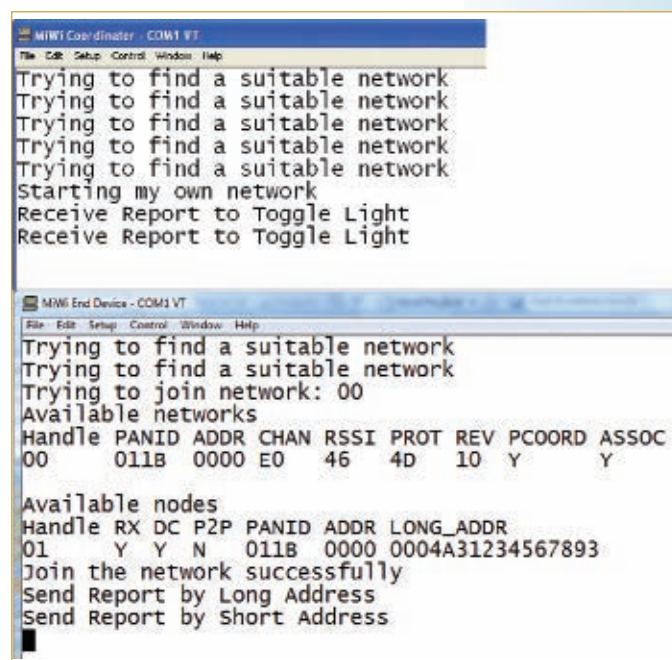
SCREENSHOT 4. About all this tells us is that the radios are speaking to each other. I think it's time to load and run some network firmware.

positive Association Response is returned to the MiWi End Device from the MiWi Coordinator and the data chase is on. The MiWi End Device will continually send Data Request frames to the MiWi Coordinator.

The **Screenshot 8** ZENA capture shows that I finally can press a button. The Send Report messages in the MiWi End Device Tera Term Pro capture say that two reports were sent. One report was sent using the MiWi Coordinator's short address and the second report was sent using the MiWi Coordinator's long address. However, the two reports in **Screenshot 8** are identical. The reason for this is that the MiWi MAC is programmed to send reports using the least amount of cost to the MAC. Thus, short addressing is a bit less work than long addressing, and both reports were subsequently sent using short addressing.

Take another look at **Screenshot 8**. I pulled the following definitions from the MiWi.h file:

```
#define USER_REPORT_TYPE 0x12
#define LIGHT_REPORT 0x34
#define LIGHT_TOGGLE 0x55
#define LIGHT_OFF 0x00
#define LIGHT_ON 0xFF
```



SCREENSHOT 5. These screens are generated as the PAN nodes are doing their things. All of this data corresponds to the fields of data in the ZENA capture.

Frame	Time(us)	Len	MAC Frame Control				Seq	Dest	Dest	Beacon	FCS																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
00001	+3373504	10	Type	Sec	Pend	ACK	IPAN	CHD	N	N	N	N	RSSI	Corr	CRC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
	+3373504							0x95	0xFFFF	0xFFFF		-02	0x65	OK																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
Frame	Time(us)	Len	MAC Frame Control				Seq	Dest	Dest	Beacon	FCS																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
00002	+1019456	10	Type	Sec	Pend	ACK	IPAN	CHD	N	N	N	N	RSSI	Corr	CRC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
	+4392960							0x96	0xFFFF	0xFFFF		-10	0x61	OK																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
Frame	Time(us)	Len	MAC Frame Control				Seq	Dest	Dest	Beacon	FCS																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
00003	+1020112	10	Type	Sec	Pend	ACK	IPAN	CHD	N	N	N	N	RSSI	Corr	CRC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
	+5413072							0x97	0xFFFF	0xFFFF		-09	0x65	OK																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
Frame	Time(us)	Len	MAC Frame Control				Seq	Dest	Dest	Beacon	FCS																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
00004	+1020432	10	Type	Sec	Pend	ACK	IPAN	CHD	N	N	N	N	RSSI	Corr	CRC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
	+6433504							0x98	0xFFFF	0xFFFF		-05	0x64	OK																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
Frame	Time(us)	Len	MAC Frame Control				Seq	Dest	Dest	Beacon	FCS																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																														
00005	+3344236	10	Type	Sec	Pend	ACK	IPAN	CHD	N	N	N	N	RSSI	Corr	CRC																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
	+39875840							0x9A	0xFFFF	0xFFFF		-10	0x66	OK																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
Frame	Time(us)	Len	MAC Frame Control				Seq	Source	Source	SuperFrame Specification																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															
00006	+4048	16	Type	Sec	Pend	ACK	IPAN	BCN	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N

SCREENSHOT 6. This is what's going on at the business end of the 802.15.4 network after I powered up the MiWi Coordinator. Here, the MiWi Coordinator is searching for a pre-existing network with no joy and establishing its own network.

Frame	Time(us)	Len	MAC Frame Control	Seq Num	Dest PAN	Dest Addr	Source PAN	Source Address	Association Request				FCS	
00007	+511072 =40390960	21	Type Sec Pend ACK IPAN CMD N N Y N	0x0B	0x011B	0x0000	0xFFFF	0x5501020304050607	Alloc Sec RxOn Power Dev AltCoord	Y N Off Batt END N	RSSI Corr CRC	-12 0x64 OK		
Frame	Time(us)	Len	MAC Frame Control	Seq Num	FCS									
00008	+1312 =40392272	5	Type Sec Pend ACK IPAN ACK N N N N	0x0B	RSSI	Corr	CRC	-06 0x62 OK						
Frame	Time(us)	Len	MAC Frame Control	Seq Num	Dest PAN	Dest Addr	Source Address	Data Request	FCS					
00009	+492016 =40884288	18	Type Sec Pend ACK IPAN CMD N N Y Y	0x0C	0x011B	0x0000	0x5501020304050607		RSSI	Corr	CRC	-12 0x63 OK		
Frame	Time(us)	Len	MAC Frame Control	Seq Num	FCS									
00010	+1152 =40885440	5	Type Sec Pend ACK IPAN ACK N Y N N	0x0C	RSSI	Corr	CRC	-04 0x65 OK						
Frame	Time(us)	Len	MAC Frame Control	Seq Num	Dest PAN	Destination Address	Source PAN	Source Address	Association Response		FCS			
00011	+5424 =40890864	29	Type Sec Pend ACK IPAN CMD N N Y N	0x0A	0x011B	0x5501020304050607	0x011B	0x0004A31234567893	Status Address	Success 0x0081	RSSI Corr CRC	-03 0x61 OK		
Frame	Time(us)	Len	MAC Frame Control	Seq Num	FCS									
00012	+1744 =40892608	5	Type Sec Pend ACK IPAN ACK N N N N	0x0A	RSSI	Corr	CRC	-12 0x63 OK						
Frame	Time(us)	Len	MAC Frame Control	Seq Num	Dest PAN	Dest Addr	Source Addr	Data Request	FCS					
00013	+1136720 =42029328	12	Type Sec Pend ACK IPAN CMD N N Y Y	0x0D	0x011B	0x0000	0x0081		RSSI	Corr	CRC	-11 0x65 OK		

SCREENSHOT 7. Once the MiWi Coordinator declared itself King, I fired up the MiWi End Device. As you can see, the MiWi End Device sensed the King's new domain and wants to associate with (join) the MiWi Coordinator's network.

Frame	Time(us)	Len	MAC Frame Control				Seq Num	Dest PAN	Dest Addr	Source Addr		FCS														
00271	+414192	12	Type	Sec	Pend	ACK	IPAN	Cmd	N	N	Y	Y				RSSI	Corr	CRC								
	+170275648								0x8E	0x011B	0x0000	0x0081	Data Request			-05	0x66	OK								
Frame	Time(us)	Len	MAC Frame Control				Seq Num	FCS																		
00272	+848	5	Type	Sec	Pend	ACK	IPAN	ACK	N	Y	N	N				RSSI	Corr	CRC								
	+170276496								0x8E				-03	0x65	OK											
Frame	Time(us)	Len	MAC Frame Control				Seq Num	Dest PAN	Dest Addr	Source Addr	Hops	ACK	Frame Control	Dest	Dest Addr	Source PAN	Source Addr	Seq Num	Report	Data	FCS					
00273	+4576	25	Type	Sec	Pend	ACK	IPAN	DATA	N	N	Y	Y														
	+170281072								0x8F	0x011B	0x0000	0x0081	0x04	N	Y	N	0x011B	0x0000	0x011B	0x0081	0x20	0x12 0x34	0x55	RSSI	Corr	CRC
																							-09	0x64	OK	
Frame	Time(us)	Len	MAC Frame Control				Seq Num	Dest PAN	Dest Addr	Source Addr		FCS														
00455	+225072	12	Type	Sec	Pend	ACK	IPAN	Cmd	N	N	Y	Y				RSSI	Corr	CRC								
	+260147856								0xEA	0x011B	0x0000	0x0081	Data Request			-08	0x64	OK								
Frame	Time(us)	Len	MAC Frame Control				Seq Num	FCS																		
00456	+832	5	Type	Sec	Pend	ACK	IPAN	ACK	N	Y	N	N				RSSI	Corr	CRC								
	+260148688								0xEA				-10	0x63	OK											
Frame	Time(us)	Len	MAC Frame Control				Seq Num	Dest PAN	Dest Addr	Source Addr	Hops	ACK	Frame Control	Dest	Dest Addr	Source PAN	Source Addr	Seq Num	Report	Data	FCS					
00457	+3632	25	Type	Sec	Pend	ACK	IPAN	DATA	N	N	Y	Y														
	+260152320								0xEB	0x011B	0x0000	0x0081	0x04	N	Y	N	0x011B	0x0000	0x011B	0x0081	0x21	0x12 0x34	0x55	RSSI	Corr	CRC
																							-08	0x63	OK	

SCREENSHOT 8. Even though the application sent the byte with long and short addresses, the MiWi MAC is programmed to send a report with the least cost to the MAC. Thus, the report was sent using the short address both times.

Frame	Time(us)	Len	MAC Frame Control	Seq Num	Dest PAN	Dest Addr	Source	Hops	Frame Control	Dest	Dest Addr	Source	Source Addr	Seq Num	Report	Data	FCS
00017	+5104	25	Type Sec Pend ACK IPAN DATA N N Y Y	0x8E	0x011B	0x0081	0x0000	0x04	ACK INTRA SEC N Y N	0x011B	0x0081	0x011B	0x0000	0x99	0x12 0x34	0x55	RSSI Corr CRC +00 0x66 OK

SCREENSHOT 9. From source and destination addresses in this ZENA capture, we can see that the MiWi Coordinator (0x0000) has sent a report to the MiWi End Device (0x0081).

Do you see the definitions in the **Screenshot 8** capture? Here's a code snippet from the MiWi application main.c file that tells us how the MiWi stack parses the definitions we just exposed in the MiWi.h file:

```
switch(*pRxData++) //report type
{
    case USER_REPORT_TYPE:
        switch(*pRxData++) //report id
        {
            case LIGHT_REPORT:
                switch(*pRxData++)
                {
                    case LIGHT_ON:
                        LED_2 = 1;
                        break;
                    case LIGHT_OFF:
                        LED_2 = 0;
                        break;
                    case LIGHT_TOGGLE:
                        LED_2 ^= 1;
                }
            }
        }
}
```

```
ConsolePutROMString((ROM char*)"Received Report to Toggle Light\r\n");
```

```
break;
}
break;
}
```

The code trail we've followed enables us to associate the report payload data byte (0x55) in the ZENA trace to the LIGHT_TOGGLE command which is parsed from the report frame that was transmitted by the MiWi End Device. The MiWi End Device LED toggle code is duplicated in the MiWi Coordinator. I generated **Screenshot 9** with a MiWi Coordinator button press.

No Pointy Hat Required

Microchip has taken the complexity out of 802.15.4 networking. With what we've discussed today, you have the ability to eliminate the wires between your robot's internal subsystems and monitor sensors, and can even build a wireless burglar alarm using MRF24J40MA modules and the MiWi stack. It's a wireless world, and you and your mechanical animal should be playing in it. **SV**

Trade Two PWM Channels for a Whole Bunch of Switches

By Jim Miller

PWM Data Channel

One day, I was working with some PWM circuits for another project and it dawned on me that even though an R/C radio sends an analog signal, with proper timing one should be able to transmit discreet data elements over a channel or a couple of channels. My first approach was to simply encode a byte on an analog channel with switches. A digital-to-analog circuit (DAC) at the controller would build the byte, then at the receiver an analog-to-digital circuit (ADC) would decode it into bits. Well, this plan sounded great but failed on two levels. I could never get the noise out of the DAC at the transmitter. Any wire I connected to the transmitter became an antenna and injected huge noise into the system.

It also had the negative aspect of robbing the controller of one control axis on a joystick. You see, the switch channels on the transmitter are binary. One cannot simply replace a switch with a potentiometer and get another analog channel. Analog is there, but the transmitter software keeps it so you can't get at it electronically. So, the project got shelved for several years in frustration. A few weeks ago, I ran across one of the prototypes and I decided I would conquer the problem with a new approach. If it worked, it would solve a big control issue for me, so I sat a while and thought. On my six-channel system, two of the channels can be controlled as binary ON/OFF modes. Why not use one for data and one for timing? Then, these two channels become a synchronous data stream! Voilà, it worked!

So now, I'm going to tell you how to trade two of your PWM channels for a whole bunch of ON/OFF switches and data. You're going to have to do a little surgery on the transmitter to intercept the switches. This article is based on a JR-XP6102 six-channel radio and receiver; I'm sure other brands will work similarly.

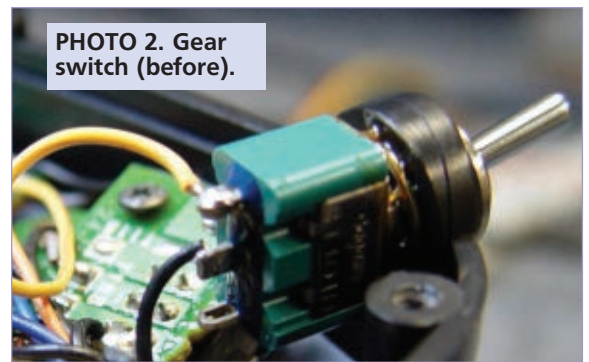
Theory of Operation

To get this theory off the ground, we should start with a quick review of PWM at the servo level. We'll start at the robot end where the PWM signal comes out of the receiver.

Pulses flow continuously out of each receiver channel. On each channel, a 0 to 1 transition happens about every 10 to 20 microseconds, depending on your brand of R/C



I like to build rovers, but communication with them has always seemed like a dicey proposition for real-time control. Two-way data radios typically have short ranges for mobile applications or are very pricey. Packetized systems tend to get glitchy or don't like the whole 'mobile' thing very much as packets get backed up and real time changes to queue time. R/C radios are fantastic for this purpose (mostly because they are designed for it ... duh). They are a very reliable and mature technology with little, rugged low-current receivers. The BIG problem with them in robotics applications is that they only carry a few analog data signals. If you want to do more than five or six channels of analog, you have to set up an entire auxiliary communications link via XBee, WiFi, BlueTooth, or some other medium. That is massive overkill if you just want to be able to control a few switch channels like lights, cameras, pumps, or select different operating modes like manual, autopilot, hover, etc. So, what do you do?



hardware and the values of other channels. The pulses do not all come out at the same moment, however. They are staggered. On the JR, it's sequentially in channel order. The staggering is not necessary electronically but nice for current loading on the servos so they don't all jump at the same microsecond.

Let's jump from the servo all the way back to the joystick on the R/C transmitter. The joystick is attached to a potentiometer which creates a voltage proportional to the stick deflection. Through the magic of electronics, this voltage disappears into a set of converters and time multiplexing circuits, then out the transmitter on an RF signal. Your receiver picks up the signal, sorts it all out, and generates the PWM pulses out of the appropriate channel.

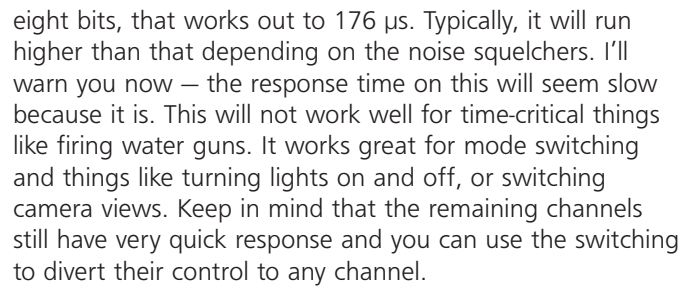
So, this whole system changes the voltage at a pot on the transmitter into a sequence of PWM pulses on the receiver. On my JR transmitter — and most other six or greater channel systems — there are channels that can be devoted to a binary type signal. I say ‘can be’ because you have to set up the transmitter to do it. These channels can also be slaved to other analog control channels. For example, in a helicopter configuration, the throttle channel might be slaved to the collective so as the blade angle of attack increases, the throttle is opened up to add power.

However, for a simple airplane configuration these channels could be assigned to the gear and flap control. The function of these controls is to extend and retract the landing gear or flaps of an R/C plane. We typically don't put landing gear halfway down or partially up, so they really only need two positions. They are driven by a two-position toggle switch on the transmitter. With only two positions, the PWM signal coming out of the receiver will only have two data segment lengths: long and short. Those are now 1 and 0.

The transmitter functions as a free-running loop that



As it turns out, the sample speed is different for different transmitters and modulation types. The speed is an integer multiple of the cycle time for all of the transmitter's channels. On my JR, that's about 22 μ s. For



We need to find control lines to hijack. The JR-XP6102

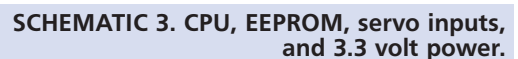
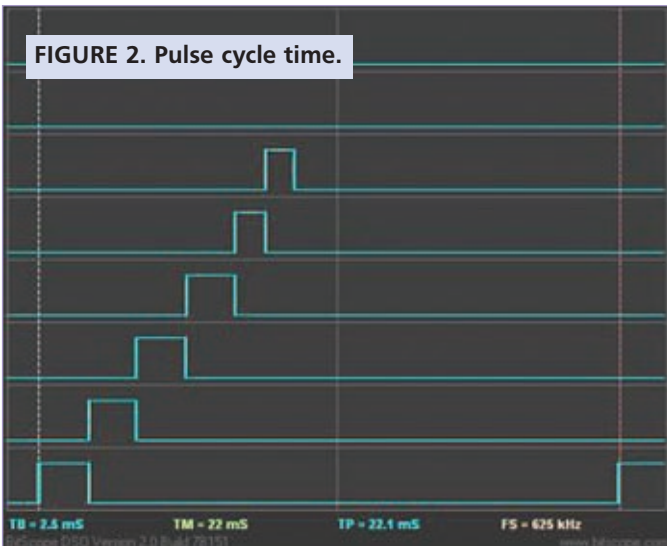


FIGURE 2. Pulse cycle time.



Connecting the Switches to the Encoder

The switch encoder is a very simple circuit and software piece. It's especially easy for a processor with the capabilities of the Parallax Propeller. The reason I chose the Prop for this was that it has the speed to make this work in its SPIN language. (Besides, I knew I could always make it work with the assembly language.) It's also my processor of choice these days; I'm comfortable with its design, programming, and the forum is an incredible asset that stays up all night long. The Propeller also has the ability to generate an NTSC signal by adding four resistors; this makes debugging a breeze.

So, we've hijacked two of the transmitter control channels. One will be a data channel and one will be a framing pulse which will allow the data channel to be properly aligned at the other end. Now you need to decide how long your data sequence will be. How many bits in a group? For the switch panel and chassis shown here, there is only room for seven switches so I chose a byte (or eight bits). The software can toggle the extra bit on every other frame as a stay-alive and error checking function. For framing, a sync pulse is sent out on the gear channel 5 before the first bit of data. This lets the receiver/decoder know when the start of a sequence of data bits is about to occur. The switch encoder software counts the cycle periods. Every time it sees 'n' periods, it toggles a software shift register to send out a data bit on the aux channel.

The decoder at the receiver end watches the framing channel 5. When it sees a high or long pulse, it captures the next eight pulses on channel 6 as either long/high or short/low. Because the bits can stretch over multiple cycles, each bit is followed by a couple of zeros.

How Long is a Cycle?

To measure a cycle, simply hook your 'scope up to an output of the receiver. Measure the period of one channel. In case you weren't listening in your physics class, a period runs from the beginning of one pulse to the beginning of the next. In **Figure 2**, it is the time from the blue-dotted vertical line on the left to the time of the red-dotted vertical line on the right, or 22.1 μ s. **Figure 2** shows all six channels but you only need to look at one. It is also important to make sure your transmitter is ON when you make the measurement. On a JR receiver, these pulses are all over the place until it syncs to a transmitter.

Transmitter/Encoder Circuit

Since I had a lot of extra capacity with the Propeller and its 32 I/O lines, I broke out everything in the transmitter circuit. Seven ports go to sense switch positions; nine ports go to LEDs; two are designated for transmitter outputs; two are used for the EEPROM; and (just for fun and debugging) four are set up for an NTSC video output for debugging.

For space considerations, the physical layout of the

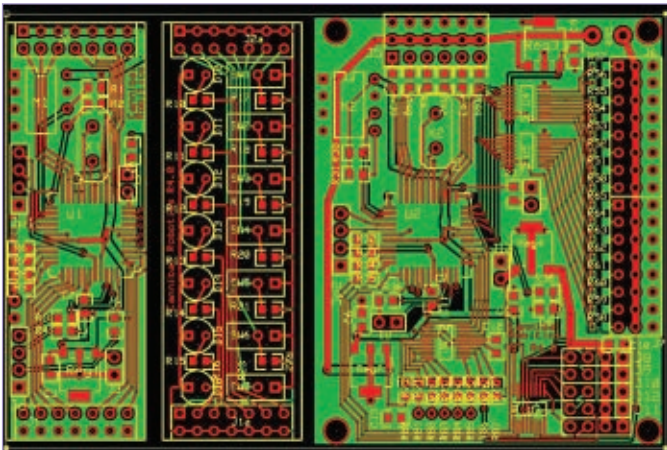


FIGURE 3. The three-part PCB.

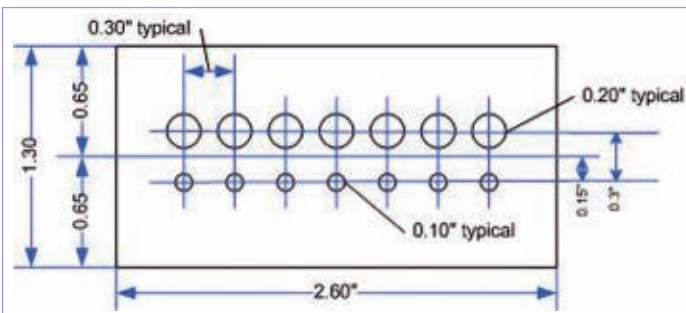


FIGURE 4. Chassis face drilling guide.

has switches on the upper left and right corners labeled "Flaps" and "Gear." When you switch these on and off, the gear and aux channel pulses from the receiver will jump back and forth between a long and short pulse. You need to find something similar to these switches on your transmitter. (I'll refer to these as the gear and aux from now on.)

To get the signal for the encoder, look at the back of the switch. On the JR, one side was grounded and the other flipped between zero and five volts, so I figured that was the signal line. I tied this to a prop port via a 10K resistor and found that it modulated the pulse length just like the switch. Okay, let's discuss the switch encoder.

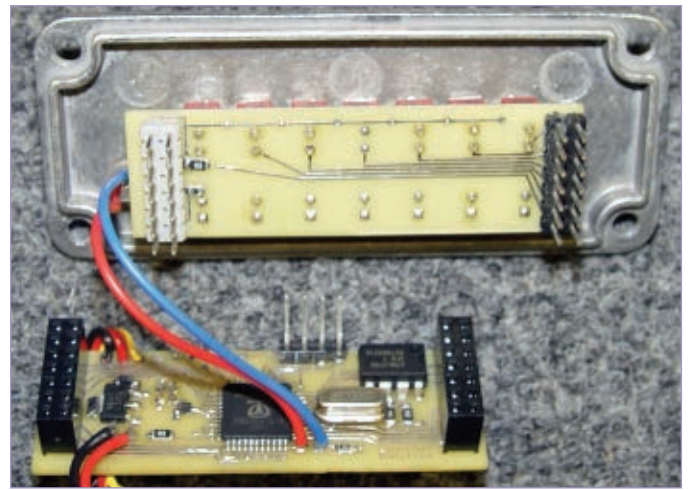
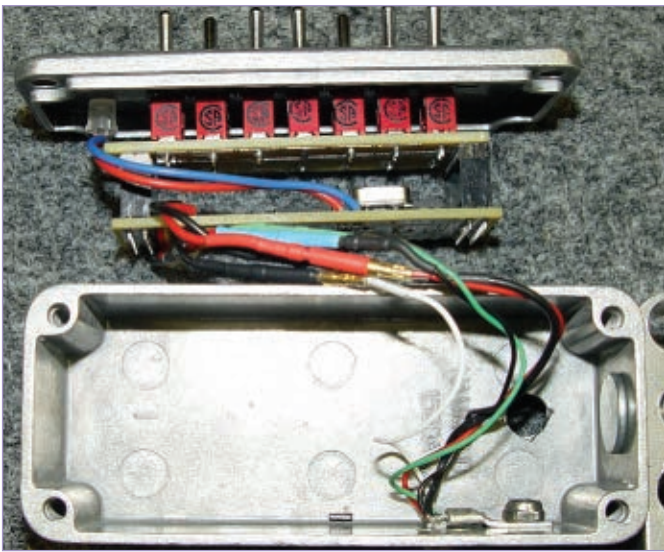


PHOTO 3 and PHOTO 4. Transmitter/Encoder case.

transmitter/encoder unfolds across two PCBs (printed circuit boards). One has the CPU, power regulator, NTSC resistor network, and crystal for the processor. The other is just switches, LEDs, and resistors. I did it this way so that I could stack the switch/LED piece on top of the CPU element, so the whole package will be small enough to fit into a box on top of the transmitter case. The two PCBs are joined by DIP headers on either end. These provide electrical connections and physical support. The switches actually attach the circuit board to the chassis face.

The circuit itself is very straightforward but there are a couple of things that need to be pointed out. The two data outputs back to the transmitter are current-limited with 10K resistors to protect everybody. You'll also notice that the switch LEDs are not hooked up sequentially on the ports. This is because the ports on any side of the Propeller have a maximum current which is less than the sum of the individual ports. Putting all of the LEDs on one side would overload the Prop if they were all turned on simultaneously. Having software control of them makes it possible to do all sorts of start-up dances. And, since there's a lot of unused processing power, I tossed in a red-green LED driver cog.

We haven't talked about the receiver yet, but the two transmitter PCBs and the receiver PCB all fit on a single ExpressPCB mini-board. All the ExpressPCB files for this are on the *SERVO* website (www.servomagazine.com). The good news is you get three PCBs on one order. The bad news is it must be cut into three individual boards when you get it. I've found the easiest and cleanest way to cut PCBs is with a band saw. If you don't have access to a band saw, score them on both sides several times with a box cutter and metal ruler, then bend to break. A dremel tool with a wheel cutter works well, too. Make sure you don't cut too close to the traces on the adjacent boards. If you don't want to order these PCBs from ExpressPCB, send me an email (jim@cannibalrobotics.com). I have extras available, so I'll cut them and put them up on eBay.

For real estate reasons, there are components on both sides of the PCBs. There are also three regulators using the surface for heat dissipation on the receiver/decoder so remember this when mounting. It will need a stand-off of at least 1/4" inch for electrical isolation and cooling. The

tag for a component on the underside of the board (green) shows up on the top (red) side of the board, so you have to be careful. The bottom side components are bypass caps and half of the voltage divider resistors for the ADC.

Switch Case

The chassis for the switch can be pretty much any size

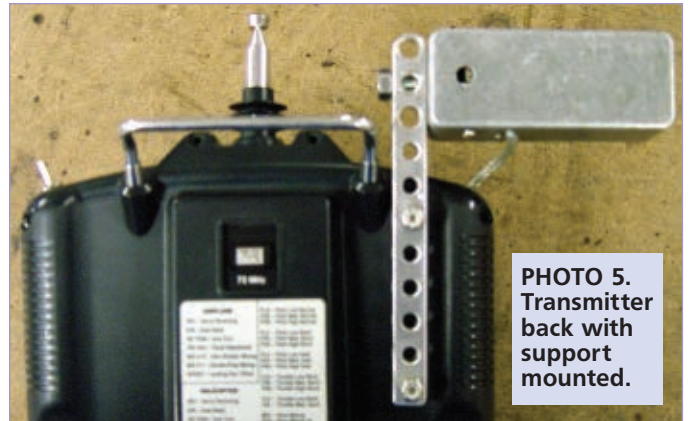


PHOTO 5. Transmitter back with support mounted.

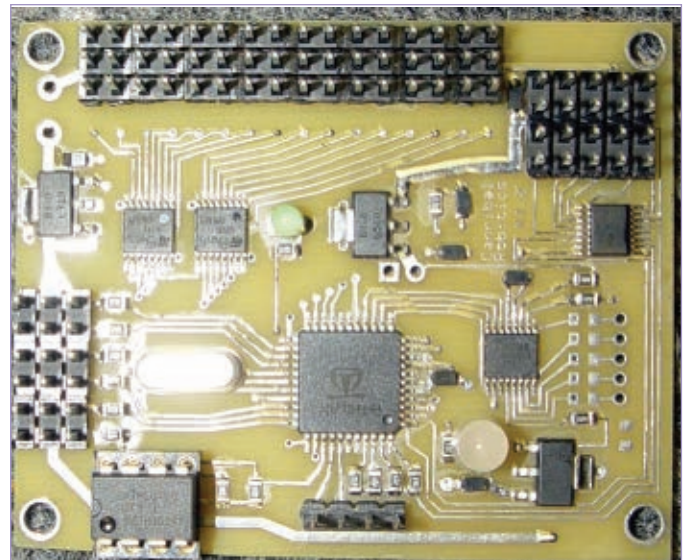
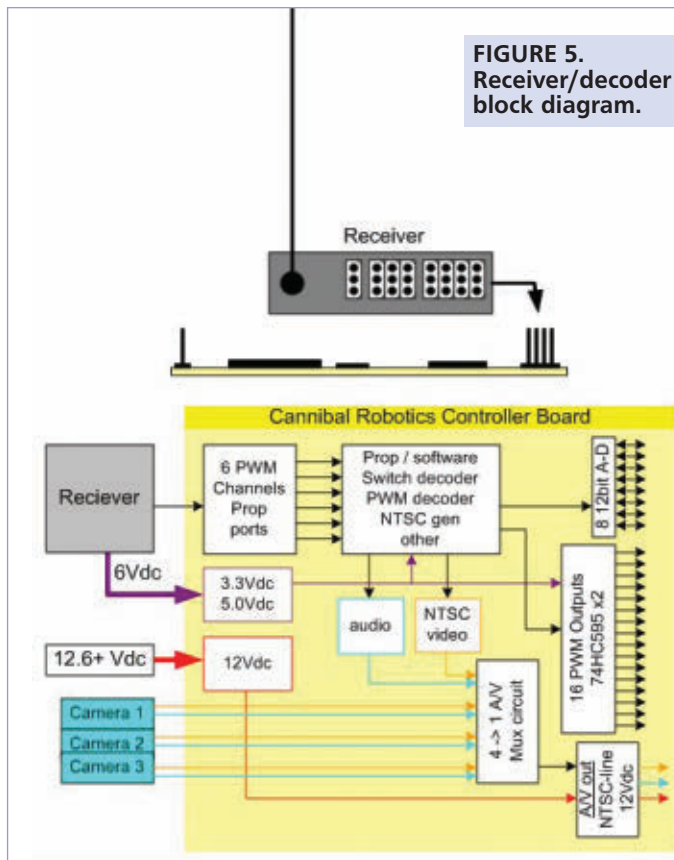


PHOTO 6. The rev 4 receiver/decoder board ready to go.

FIGURE 5.
Receiver/decoder
block diagram.



you need. I wanted it as small as possible and very light. Given the proximity to the antenna, a metal or conductive plastic enclosure is very important. You need to make sure the case is grounded to the radio ground. For power, I tapped it directly off of the battery at the on-off switch in the transmitter. You'll see that the red-green LED is wired off of the lower board and pressed into a hole on the

chassis face. Call it a lack of planning, but I just didn't want to mess up the symmetry of the switch board layout.

As a side note on those disconnected switches, JR installs SPDT switches but only uses two of the three poles with the center grounded. With some clever rewiring, one could keep those switches operational. By moving the ground to an outer pole, the transmitter sense wire to the center, and the encoder output to the other pole, the switch becomes a selector between OFF and the encoder input.

Drilling the holes in the chassis for the switches and LEDs to line up properly takes some patience and forethought. I recommend having a pattern to follow; use a punch to mark the holes before drilling. This is a tight layout and the nuts on the switch bodies come into contact when it's laid out correctly. If you miss, you may not have some of the switches tightened. The switches provide the mechanical support for the circuit inside of the box. When you solder them to the PCB, make sure they are lined up well and firmly seated on the board. You don't want them bent at a bad angle to get into the hole.

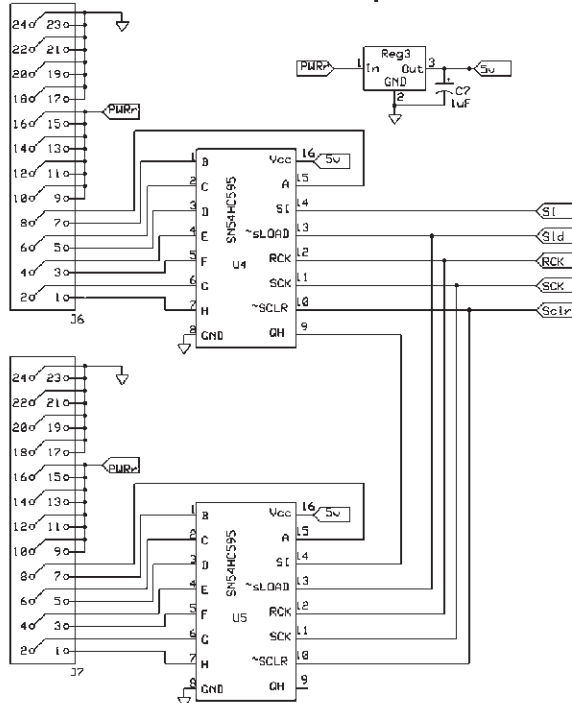
The best order of construction is to populate the resistors and DIP headers on the board; drill the chassis; then insert the switches and LEDs into the holes in the chassis; finally, position the PCB on top. Don't tighten the switch nuts or you'll have a lot of trouble getting all of the pins through the holes. Once everything is lined up, solder away. A dab of epoxy on the side of a few switches is probably not a bad idea either for extra physical support but make sure they are in the holes for alignment while the epoxy cures.

The wiring from the encoder into the transmitter case must be shielded and grounded on both ends. Make it as short as possible. I do not recommend a connector on the

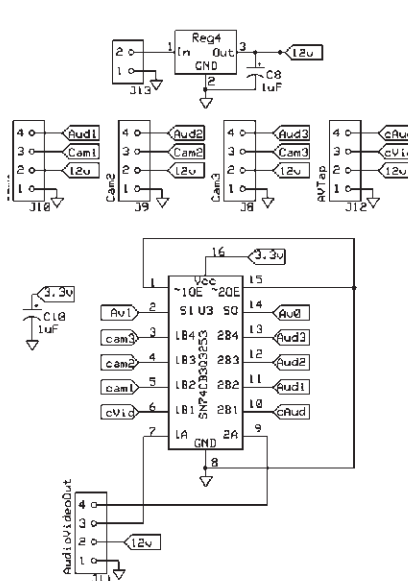
outside of the cases. You can use RS-232 pins as disconnects but make sure they are completely inside of the case. I used a section of four-conductor shielded wire cut from a retired USB hard drive cable. The shield carries the ground; the others are: power, sync, and data.

Mounting the chassis on the transmitter case will require a bit more drilling. You want it to be solid, adjustable, and close to the action. A 0.5" piece of aluminum square bar cut to about 6" in length and attached to the back of the case with two screws provides a very solid anchor. It's also positioned so that it enhances the fingertip 'grip' of the

Servo Output



Audio Video Switch



SCHEMATIC 4. Servo outputs,
AV multiplexor.

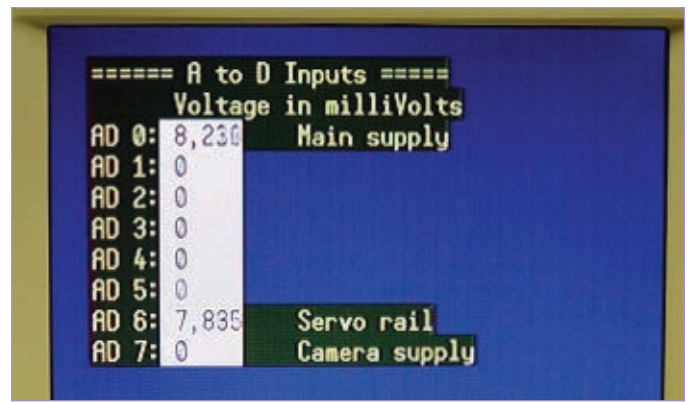
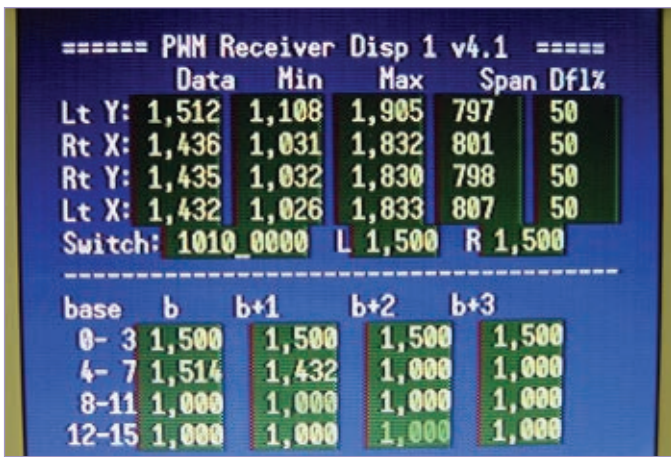


PHOTO 7 and PHOTO 8. Configuration screens.

transmitter. One 1/4" x 20 bolt 1" long attaches the encoder to the top of the square bar. Coming out of the side of the case, it provides a rotational adjustment to get it just right.

The Transmitter Software

The encoder software is incredibly simple (and well commented in the SPIN code). It counts the clock cycles. On every eighth pulse, it sends a framing marker and sets up the next data bit. Each data bit and framing marker is held in place for at least one complete cycle. Switch positions and LED status are updated at the beginning of each bit cycle. This all resides in one cog. The NTSC generation happens in another cog but unless a video display is added to the transmitter above the switches, there is not a lot of use for this beyond debugging. Since the encoder is so incredibly simple, there is not much to display. In the code, there are remarks to point the switches and LEDs to the ports you have chosen.

The red-green LED driver is fun too. It's off if you send it a zero; a number between one and 255 will give you a combination of red and green — one being all green and 255 being all red. All of the code is available in the download package on the *SERVO* website.

The Receiver Circuit

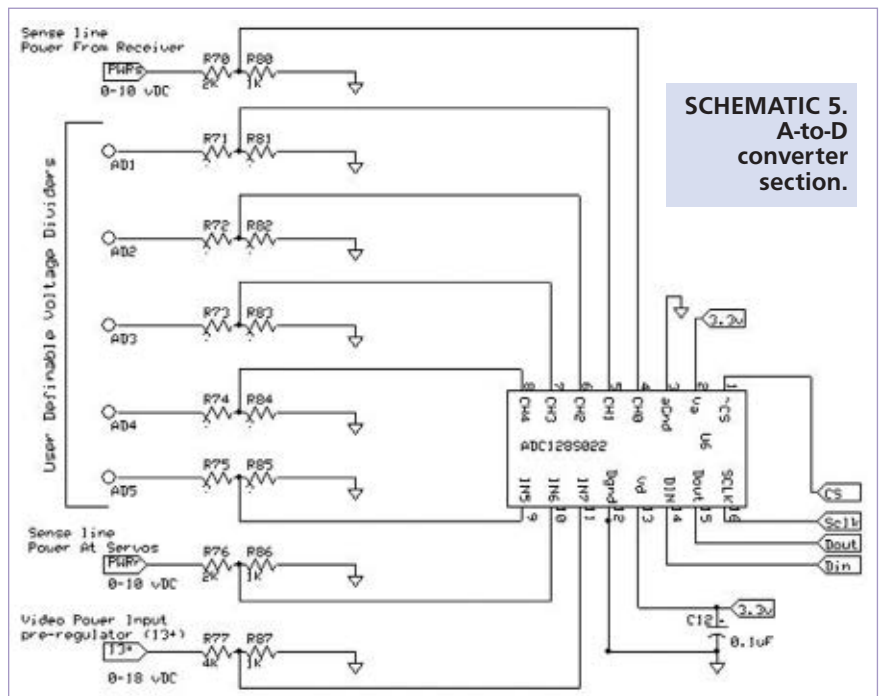
The receiver side is more complicated than the transmitter side because it handles more processing and switching. It is divided into functional groups in the **schematics** but it all resides on one board. Functionally, there is the EEPROM and CPU power, the PWM receiver inputs section, the servo output section, the audio/video switching and 12V power, an NTSC signal generator DAC, an audio output, an eight-channel ADC, and finally, some indicator lights for power and activity. The **block diagram** rounds all of this up at a higher level. The EEPROM and crystal circuits are copied directly from the Propeller demo board.

The PWM receiver inputs are all current-

limited with 10K resistors. The output level of the signals will vary with the power supply of the receiver, but they do go to zero when low. The resistors compensate for the upper voltage variance.

The audio and video switching section is facilitated by a SN74CB3Q3253 2 x 4 to one analog multiplexor. One of four separate A/V inputs can be selected and routed to a transmitter or the tap connector. Channel 0 is hard-wired to the audio and video outputs of the Propeller and is the default position of the selector. Channel selection is accomplished electronically by the bit configuration of Av0 and Av1.

The servo output section of the board is pretty cool. Two 74HC595 serial-in, parallel-out shift registers have been cascaded into 16 outputs. These outputs drive the signal pins on 16 servo connectors. (It takes five Propeller ports and a little PASM code to create 16 PWM output channels.) You'll note that the power rail for the servo connections is routed from the receiver through a one ohm, five watt resistor, R45. This provides a voltage difference between the supply and the power rail which facilitates a current calculation for the servos. (The math is noted in the SPIN



SCHEMATIC 5. A-to-D converter section.

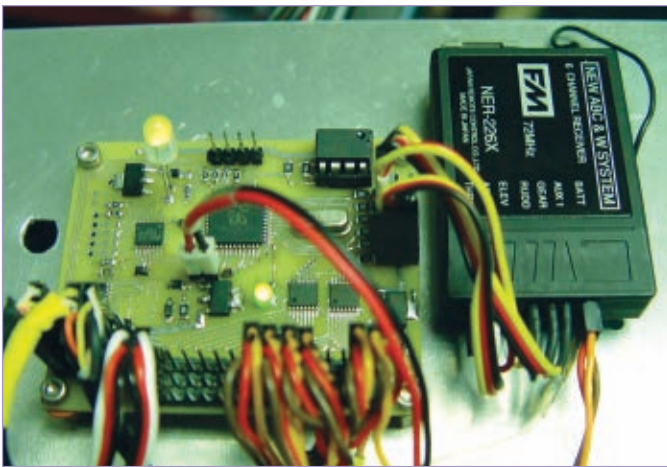


PHOTO 9. Revision 4 receiver/decoder board.

code comments.) That way, you can calculate holding if you leave a servo to hold something in a certain position. You can also determine when peak loads might be happening and reorder your events to reduce battery drain.

There is a 12-bit, eight channel ADC on the board. This reads 0 to 3.3 volts, or the exact supply level of your voltage regulator on every channel. It is hard-wired to sample the power level inputs from the receiver and video power source. It's designed primarily to watch battery levels. It also watches the output voltage on the servo power rail. All inputs to the A-to-D channels are routed through voltage dividers on the board. The base resistors are all 3.3K and the sample resistors can be adjusted to accommodate your desired voltage range. (That's why several of the resistor values are question marks in the schematic.)

In the code comments, I've noted some of the common resistor values for voltage ranges in the code. To make it simple, the base resistor is 3.3K to match the supply 3.33 volt level. I've also provided the divisors for the 12-bit sample for each value in the code. The remaining inputs are good for sampling battery voltages, sensor outputs, or pot positions.

This implementation of the ADC is not the optimum. There is noise on the power supply and analog reference line. In the ADC128S022 datasheet, there is a better noise canceling arrangement which would make for a quieter implementation and take better advantage of the 12-bit resolution. Due to board space restrictions, I decided to just deal with the noise by sacrificing some resolution and averaging.

The Receiver/Decoder Software

On the receiver/decoder side, the software is a bit more complex, but takes advantage of the multi-core aspect of the Propeller. The frame and data channel (aux and gear) are routed into a cog running SPIN code which builds and returns a byte representing the switch positions on the transmitter. The four remaining receiver channels are routed into a cog running a SPIN program available on the Parallax object exchange (<http://obex.parallax.com>) by David C.

Receiver/Decoder Parts List

Legend	Description	Digi-Key Part Numbers
C5-C7	1 μ F	511-1453-1-ND
C8	10 nF	
C9-C12	1 μ F	511-1453-1-ND
C12	0.1 μ F	
D2	Red Green LED	
D3	PWR LED	
M2	AT24C512 EEPROM	AT24C512
R30-31	10K	P10KACT-ND
R32	1.1K	P1.1KACT-ND
R33	560	P560ACT-ND
R34	270	P270ACT-ND
R35	560	P560ACT-ND
R36-37	220	P220ACT-ND
R38-44	10K	P10KACT-ND
R45	1.000 1W	
R50-65	10K	P10KACT-ND
R70	6.8K	P6.8KACT-ND
R71-75	see text	
R76	6.8K	P6.8KACT-ND
R77	30K	P30KACT-ND
R80-87	3.3K	P3.3KACT-ND
Reg2	3.3V regulator	LM3940IMP-3.3CT-ND
Reg3	5V regulator	LM3940IMP-5.0CT-ND
Reg4	12V regulator	LM3940IMP-12CT-ND
U2	Propeller processor	P8X32A-Q44-ND
U3	SN74CB3Q3253	296-17283-1-ND
U4-5	74HC595	497-7387
U6	ADC128S022	ADC128S022C1MT-ND
X2	5 MHz	X158-ND

Gregory. This returns the width of the signals in microseconds for each channel. A third cog is used to run my assembly routine which provides the PWM signals to the 16 servo outputs.

With these cogs running, the main program has the numerical input data from the four transmitter joysticks and the switch position information. It can manipulate and route that joystick data to any or all of 16 different outputs. This can be a lot of fun. Servos can be adjusted and held in a position while a control is shifted to another servo. Servos can be ganged or slaved to other inputs, or switched from autopilot to manual control modes. The AV switching software is about as simple as it gets and is handled as a single PUBLIC routine in the main program. Send it a camera number between 0 and 3, and it sets the bits accordingly. The RG object is actually used on both the receiver and transmitter sides to drive the red-green status indicator LED.

Finally, a cog is used to drive the NTSC video output. On the decoder side, this is very useful as you can watch values change and see exactly what you are working with. In the downloadable code, I have provided some well formatted screens that will help you set up your software.

Transmitter Parts List

Legend	Description	Digi-Key Part Numbers
C1-C4	1 μ F	511-1453-1-ND
D1	LED power	160-1142-ND
D10-16	LED switch	160-1142-ND
J1	14-pin socket	S7075-ND
J2	14-pin socket	S7075-ND
J3	Two-pin header	929710-10-36-ND
J1a	14-pin header	929710-10-36-ND
J2a	14-pin header	929710-10-36-ND
Jp1	Four-pin header	929710-10-36-ND
M1	EEPROM	AT24C512
R1-R4	10K	P10KACT-ND
R5	220K	P220KACT-ND
R6	1.1K	P1.1KACT-ND
R7	560	P560ACT-ND
R8	270	P270ACT-ND
R9	560	P560ACT-ND
R10-R16	220	P220ACT-ND
R17-23	10K	P10KACT-ND
Reg1	3.3V regulator	LM3940IMP-3.3CT-ND
SW1-SW7	PC mount toggle switch	RadioShack 275-645
U1	Propeller processor	P8X32A-Q44-ND
X1	5 MHz	X158-ND
	PCBs, chassis, solder, etc	

The Code

The downloadable code comes with some formatted displays which show the inputs and outputs of the system, and facilitates getting a feel for everything. **Photos 7 and 8** show configuration screens. The displays present the various data elements, switch positions, and raw PWM outputs. The code has some good examples of how to use the switch data for different modes and control. It has a tractor drive subroutine to turn X-Y joystick positions into right and left. For my transmitter switch assignments, the last three switches control display configuration and camera selection.

The first switch is a brake that turns on/off all motion — fantastic for troubleshooting. I can also tune the value to the exact 0 point on the PWM speed controllers. Switches are used for changing the right joystick from vehicle motion to pan and tilt for cameras and sensors. I also have a switch that moves vehicle control from pure manual linear tractor drive to a PID “fly by wire” mode. Your robot design will ultimately dictate the applications, but it’s wide open when you start to ponder.

Setting Up Your Software

If you use the provided code for the base of your rover, you’ll need to set some variable limits. The TractorDrive section of the code takes the XY translation of a joystick and calculates left and right drive speeds. It uses the



PHOTO 10. Roofer with the receiver/decoder board under its front cowl. (Note the antenna placement on left front.)

deflection percentage of the joysticks — not the actual values. The SetDeflection routine calculates this using the upper and lower limits of the pulse widths. These limits are input by the programmer. You can test for these limits by using the video screen output. Most transmitters are programable for upper and lower limits. If you’re handy with transmitter programming, set them all to the same upper and lower limit timing and you can reduce the number of variables in your code.

Board Onboard

Photo 9 shows the receiver/decoder (rev 4.0) board. Note the connection to the receiver; not all channels carry all three lines. One or two beefy power and ground wires are probably plenty. The PCB is very small. I did it this way so that placing it in a project would be pretty easy. The big red/black wire is the 14 volt supply for the camera regulator. If you’re using less than 12-volt cameras, you don’t need the 12V regulator. You can route your power supply for them directly to the power rail on the connectors, or use a different value regulator. Some tiny cameras work on nine, and five and six volts. The mounting holes for the decoder board are spread out to the corners as far as possible to allow the transmitter to be mounted below it. (Remember for your application, there are other components on the back and some air flow around it is necessary.) In my application, the audio/video output is routed into a 700 MHz video transmitter. You can also use a hacked 2.4 GHz transmitter from one of the battery powered X-10 transmitters.

Photo 10 shows the first vehicle to carry my board — Roofer! By the time you read this, it will also be installed in DeckBot; check the Parallax forum for progress on that project. Once the transmitter modifications are in place and the encoder is working, this is a highly transportable concept. I hope you’re able to integrate this design into your robots. **SV**

The *SERVO* Webstore

Attention Subscribers ask about your discount on prices marked with an *

CD-ROM SPECIALS



6 CD-ROMs & Hat Special
Only \$ 129.95 (includes shipping)!

www.servomagazine.com



On Sale Now!
Only \$24.95

ROBOTICS

PIC Robotics

by John Iovine

Here's everything the robotics hobbyist needs to harness the power of the PICMicro MCU!



In this heavily-illustrated resource, author John Iovine provides plans and complete parts lists for 11 easy-to-build robots each with a PICMicro "brain." The expertly written coverage of the PIC Basic Computer makes programming a snap – and lots of fun.

\$24.95

Forbidden LEGO

by Ulrik Pilegaard / Mike Dooley

Build the Models Your Parents Warned You Against.

Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse. **\$24.95**

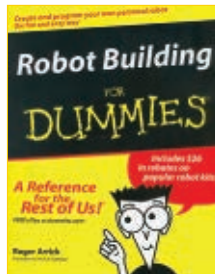


Robot Building for Dummies

by Roger Arrick / Nancy Stevenson

Discover what robots can do and how they work. Find out how to build your own robot and program it to perform tasks. Ready to enter the robot world? This book is your passport! It walks you through building your very own little metal assistant from a kit, dressing it up, giving it a brain, programming it to do things, even making it talk. Along the way, you'll gather some tidbits about robot history, enthusiasts' groups, and more.

\$24.95



Build Your Own Humanoid Robots

by Karl Williams

GREAT 'DROIDS, INDEED!

This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than \$300. Together, they form the essential ingredients for making your own humanoid robot. **\$24.95***



Robot Programmer's Bonanza

by

John Blankenship,
Samuel Mishal

The first hands-on programming guide for today's robot hobbyist!

Get ready to reach into your programming toolbox and control a robot like never before! *Robot Programmer's Bonanza* is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.

\$29.95



Robotics Demystified

by Edwin Wise

YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!

Now anyone with an interest in robotics can gain a deeper understanding – without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **\$19.95**



We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles



Published by T & L Publications, Inc.

\$57
per bundle

Save \$10
off the
normal
price!!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, and 09.

Kickin' Bot

by Grant Imahara

Enter the arena of the metal gladiators!

Do you have what it takes to build a battle-ready robot? You do now! Here are the plans, step-by-step directions, and expert advice that will put you in competition — while you have a heck of a lot of fun getting there. Grant Imahara, the creator of the popular BattleBot Deadblow, shares everything he's learned about robot design, tools, and techniques for metal working and the parts you need and where to get them.

\$24.95



How Would You Like To See Your House Cleaned?



TODAY



SERVO'S VISION OF THE FUTURE

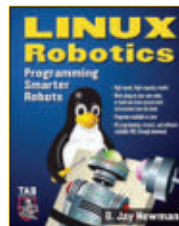
Visit my online store today! <http://store.servomagazine.com>

Linux Robotics

by D. Jay Newman

If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.

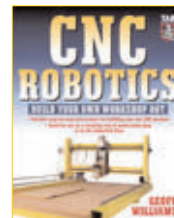
\$34.95



CNC Robotics

by Geoff Williams

Here's the FIRST book to offer step-by-step guidelines that walk the reader through the entire process of building a CNC (Computer Numerical Control) machine from start to finish. Using inexpensive, off-the-shelf parts, readers can build CNC machines with true industrial shop applications such as machining, routing, and cutting — at a fraction of what it would cost to purchase one. Great for anyone who wants to automate a task in their home shop or small business. **\$34.95**



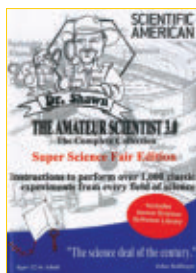
SPECIAL OFFERS

The Amateur Scientist 3.0 The Complete Collection

by Bright Science, LLC

There are 1,000 projects on this CD, not to mention the additional technical info and bonus features. It doesn't matter if you're a complete novice looking to do their first science fair project or a super tech-head gadget freak; there are enough projects on the single CD-ROM to keep you and 50 of your friends busy for a lifetime!

\$29.95



The Getting Started Combo includes: *Getting Started in Electronics* by author Forrest Mims and the DIY Electronics Kit. In his book, Mims teaches you the basics and takes you on a tour of analog and digital components. He explains how they work and shows you how they can be combined for various applications. The DIY Electronics Kit allows for the hands-on experience of putting circuits together — the kit has over 130 parts! No soldering is required and it includes its own 32 page illustrated manual.

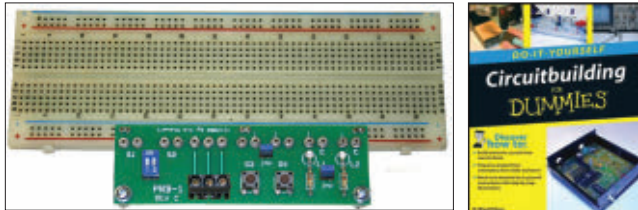
Combo Price \$62.95 Plus S/H

SPECIAL OFFERS

Proto Buddy Kit & Book Combo

For those just getting started in electronics as a hobby, a solderless breadboard (SBB) is the perfect platform for building those first circuits. Attach a Proto Buddy to an SBB, include a battery or two, and you will have a combo that has a lot of the same functionalities as more expensive units.

Combo includes PCB & Components, 830 point SBB, and Do-It-Yourself Circuitbuilding For Dummies.

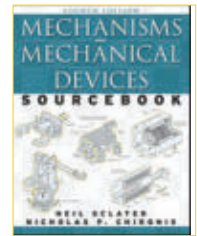


Combo Price \$57.95 Plus S/H
Limited time offer.

Mechanisms and Mechanical Devices Sourcebook,

by Neil Sclater,
Nicholas Chironis

Over 2,000 drawings make this sourcebook a gold mine of information for learning and innovating in mechanical design. Overviews of robotics, rapid prototyping, MEMS, and nanotechnology will get you up-to-speed on these cutting-edge technologies. Easy-to-read tutorial chapters on the basics of mechanisms and motion control will introduce those subjects to you. **Reg \$89.95 Sale Price \$69.95**



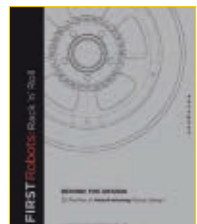
FIRST Robots: Rack 'N' Roll: Behind the Design

by Vince Wilczynski,
Stephanie Slezyski

More than 750 photographs!

The second annual book highlighting the creativity and process behind 30 winning robot designs from the 18th annual international FIRST Robotics Competition. The FIRST organization, founded by Dean Kamen (inventor of the Segway), promotes education in the sciences, technology, and engineering.

Reg \$39.95 Sale Price \$29.95



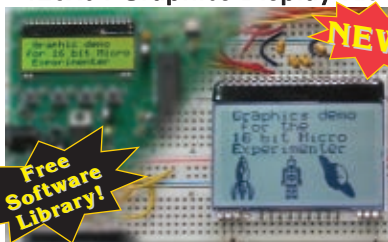
Enter the world of PICs & Programming with this great combo!

Combo Price \$175.95

For complete details visit our webstore @ www.nutsvolts.com

PROJECTS

128x64 Graphics Display Kit



New application for the 16-Bit Micro Experimenter

LCD displays ... they have been around for quite some time, but what if you could have both characters as well as graphic displays at the same time? With this kit, we will show you how easy and inexpensive this technology can be using the 16-Bit Micro Experimenter.

Subscriber's Price \$45.95
Non-Subscriber's Price \$48.95

16-Bit Micro Experimenter Board

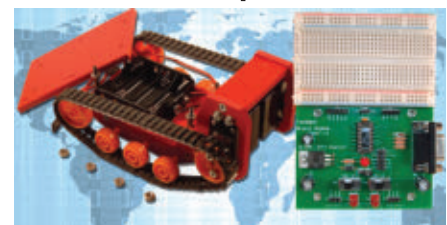


Ready to move on from eight-bit to 16-bit microcontrollers? Well, you're in luck! In the December 2009 *Nuts & Volts* issue, you're introduced to the 16-Bit Micro Experimenter.

The kit comes with a CD-ROM that contains details on assembly, operation, as well as an assortment of ready-made applications. New applications will be added in upcoming months.

Subscriber's Price \$55.95
Non-Subscriber's Price \$59.95

Tankbot Kit & Brain Alpha Kit



As seen in the Sept. issue Tankbot/ Brain Alpha
by Ron Hackett

A series filled with projects and experiments to challenge you through your learning process while you grow your fully expandable Brain Alpha PCB!

The brain is a PICAXE-14A!

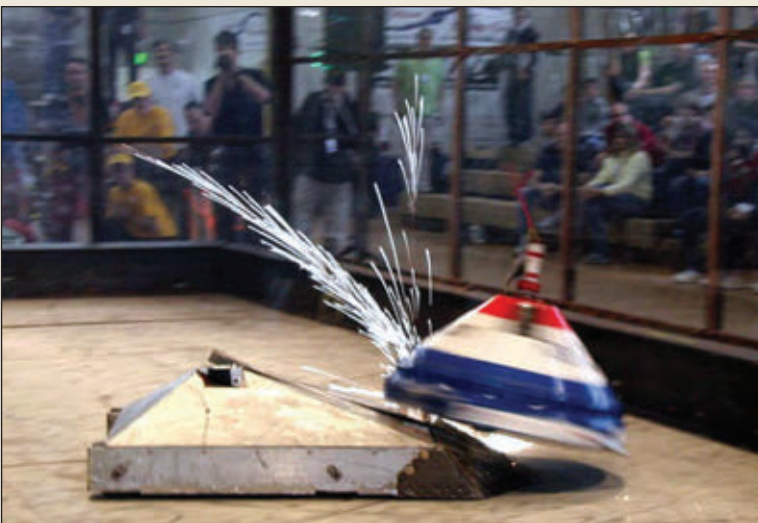
For more info & pictures, visit the *SERVO* Webstore. Tankbot and the Brain Alpha Kit can be purchased separately.

Combo Price \$ 138.95

So You Want To Build A ComBot

Part 1

By Greg Intermaggio



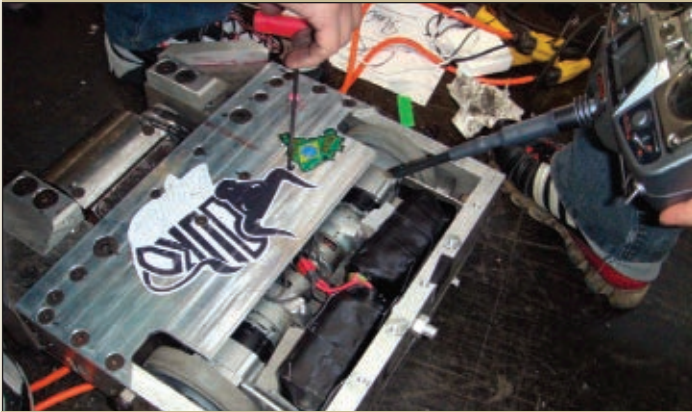
So, you've seen the fighting robots on TV, and you finally got off the couch to go to RoboGames and see a ComBot event in person. Up close, it's a whole different game. You can't tell on TV, but most of those robots are bigger than your dishwasher and made of heavy metal designed to tear their opponents to shreds. Saws spinning, hammers slamming, and flames spitting from the flame throwers, you just get drawn in. There's no other sport like it. So, you decide you want to build your first ComBot. This two-part series covers the basics of designing a 30 pound (Featherweight) ComBot from the ground up using only tools you'd find in a home shop.

In this first installment, we'll talk about what goes into building a ComBot and begin the design process. For more information on the ComBots event, visit <http://combots.net/> and see the **Sidebar** for additional resources.

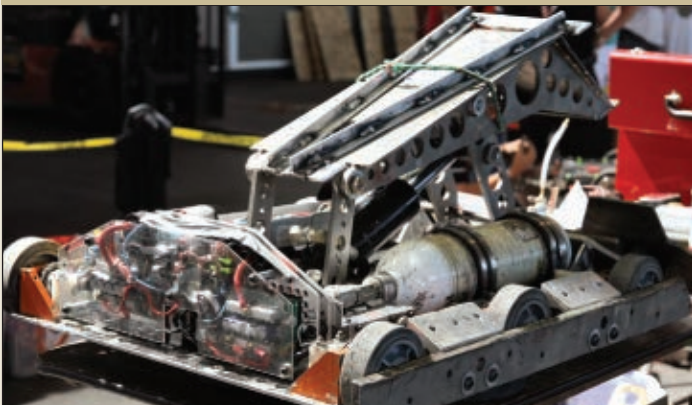
The first thing to consider in building your ComBot is cost. Robots get very expensive, very fast. Building a basic 30 pound ComBot will cost somewhere in the neighborhood of \$800-\$1,600. For a cheaper alternative, consider building a three pound bot instead, which will likely cost \$400-\$800. We'll also want to keep in mind the nature of the sport — even in the 30 pound weight class, these robots are designed to destroy each other. Robotics team Tesla Prime designed "Overnight Delivery," a 30-pound ComBot which had an upturned 1/16" steel cooking wok as armor. In its first match against Chainzilla — a 30 pounder with a spinning steel bludgeon — Chainzilla managed to literally tear holes into Overnight Delivery's steel armor, leaving the critical electronics exposed. The robot was incapacitated within just a few short seconds of the match starting. In this respect, an \$800 ComBot can quickly become a \$1,600 ComBot, and a \$1,600 ComBot can quickly become a \$3,200 ComBot. Remember that the goal of the competition is to destroy your opponent. Factor this potential for destruction into your budgeting for the project.



Team PlumbCrazy (left to right): Wendy, Pipe Wench, Sewer Snake, Matt, Devil's Plunger.



Team RioBotz works fast to fix their Combot between matches.



The infamous super-heavyweight Ziggy without his shell of armor.



Touro Maximus is an example of excellent weight distribution and well-protected wheels.

Here's what goes into a 30 pound ComBot:

- Armor and Chassis
- Speed Controllers
- Motors
- Wheels and Hubs
- Batteries
- Radio Control Transmitter
- Radio Control Receiver
- Fuse
- LED Indicator Light and Resistor
- Master Power Switch
- Wires

How does it all fit together? Here's a ComBot in a nutshell:

- The master power switch is flipped, allowing electricity to flow from your batteries through a fuse which protects your circuits from a short.
- The electricity flows from the fuse to your speed controllers.
- At the same time, different batteries supply power to the receiver which receives signals from the transmitter.
- The receiver translates the radio signal from the transmitter into an electrical signal which is sent to the speed controllers.
- The speed controllers interpret the electrical signal from the receiver and use it to determine how much voltage to release to the motors.
- The motors receive the voltage and start spinning, causing the gear train to start spinning and, as a result, spinning the wheels.
- The bot then moves forward at a speed controlled by the transmitter.
- The chassis holds everything together and the armor protects everything from harm (like when the bot smashes into a wall at full speed!).



Sewer Snake — Team PlumbCrazy's heavyweight behemoth — being tipped by Matt Maxham and a friend from RioBotz.

ARMOR and CHASSIS

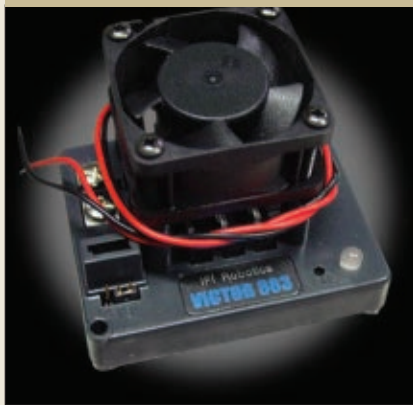
Arguably, the most important aspect of a ComBot is the armor that protects it from the attacking opponents. Next in line will be the chassis that holds everything together. For most applications, steel or aluminum will offer the best combination of affordability, durability, and weight. Bots with heavier active weapons like hammers or large saw blades may want to conserve weight by using aluminum, where bots with weight to spare may opt for steel.

BotTip: Always aim to have the lowest center of gravity possible. This way, the Combot will be harder to invert and easier to control.

SPEED CONTROLLERS

Speed controllers are about as close to a brain as you'll find in a ComBot. The purpose of a speed controller is to interpret signals from the receiver, and turn those signals into voltage to your motors. We'll need one speed controller; per motor for this tutorial, we'll use an IFI Victor 883 speed controller.

IFI Victor 883 speed controller.

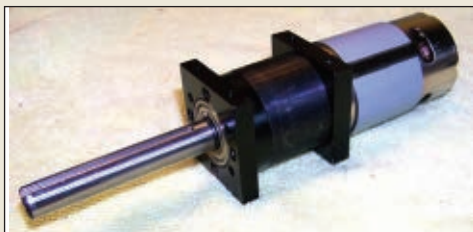


IMPORTANT: For this application of the IFI Victor 883 speed controller, we'll need signal boosting cables rather than the ones that are included. Signal boosting cables are generally available as an accessory and are used to amplify the signal sent from the receiver. This allows the speed controllers to have more accuracy. The 883 has three connections (not including the fan): the battery input; receiver input; and motor output.

BotTip: You'll want to triple-check that your wiring is perfect before putting any power through your speed controller. If the unit short circuits or the battery polarity is reversed, the controller will be destroyed.

MOTORS

Picking the right motor for your ComBot is essential. If you choose a motor with too little torque, your bot won't move. If you choose a motor with too much torque, it will



Magnum 775 planetary gearmotor.

tear your robot apart the first time you run it. We'll be using the Magnum 775 planetary gearmotor. This will give us plenty of

torque to work with while maintaining a light weight and relatively low battery consumption. You'll need two drive motors, and may want to consider buying a third motor for an active weapon. For beginning ComBot builders, it's easiest to start small by creating a wedge bot with no active weapon.

BotTip: Indirect drive (using gears, chains, or belts to connect your motors to your wheels) is an excellent method for protecting your motor shafts from heavy impacts that could potentially ruin your motors. For the sake of simplicity, we won't cover specifics of creating an indirect drive system, but if you are mechanically inclined, it's a highly favorable option!

WHEELS and HUBS

Having the right wheels for your robot can mean the difference between slamming your opponent into the arena walls at full speed or spinning out when you take a sharp turn.

Colson wheels are a great mix of traction, low friction, light weight, and durability. Depending on the diameter of wheel you select, you may have a different shaft size. In most cases when working with Colson wheels, you're best off just buying a hub and drilling directly into the side of the wheels to connect them.

Colson 4x1 and a quarter wheel.



BotTip: Protect your wheels! Plan to have your wheels inside the body of your bot so that the only visible parts of the wheels are the parts touching the ground to move the robot.

RADIO CONTROL TRANSMITTER

The transmitter is the device that sends the radio signal telling your ComBot what to do. For our ComBot, we'll be using a Spektrum DX6i six-channel 2.4 GHz radio system mode 2. This will give us plenty of channels to work with for drive motors and weapons, while eliminating all interference from other radios.

Transmitter.



RADIO CONTROL RECEIVER

The radio control receiver will receive the signals from the transmitter, and send them to the speed controllers to be converted into voltage to the motors. The Spektrum BR6000 six-channel bot receiver is the perfect choice for this ComBot since it is not only compatible with the transmitter, but also has a fail-safe on all channels (which means it will be combat-legal out of the box). Additionally, a special battery is needed to power the receiver.



Spektrum BR 6000 receiver.

BotTip: If you decide to use a Spektrum AR model, you'll need to purchase a mechanical fail-safe for the throttle channel. This is because receivers that identify a throttle channel fail-safe as an "idle" position rather than an "off" position by default are illegal at most combat robotics events. (If your bot loses communication with your transmitter, you certainly don't want your saw blades idling while you try to turn it off!)

BATTERIES

Batteries are one of the trickiest components of a ComBot. This is because you want just the right balance between power and longevity in the lightest package possible. To put things into perspective, most ComBots will be running slower at the end of their matches than when they started. In exhibition matches with no time limit, the winner is sometimes determined by who runs out of battery power last. For this bot, we'll use a single 18V 3.0 Ah NiCad BattlePack.



BattlePack battery.



Receiver battery.

BotTip: Interested in reading advice and insight from some of the top ranked ComBot builders in the sport, as well as getting updates on cool things going on at ComBots? Go to **SuicideBots.com** for hot bot-on-bot action!

FUSE

A fuse is an extremely important component that should be in every ComBot. It's often the only thing standing in the way of a complete electrical failure, and can mean the difference between a \$20 replacement as opposed to a \$2,000 replacement. The job of the fuse is simple: Be the weakest link in the electrical system of the robot, so that if there's a short circuit, the circuit loses power before any critical components are damaged. You can grab a good fuse at your local electronics store. For our featherweight ComBot, we'll use a 60 amp fuse. (It's a big boy!)



Fuse holder.



60 amp fuse.

LED INDICATOR LIGHT and RESISTOR

Most combat robotics events require that contestants have a highly visible LED on their robots to distinguish whether the power is on or off. This just means adding a simple circuit with an LED and a resistor — more on this later.



Resistor.



LED.

MASTER POWER SWITCH

Every ComBot needs a master power switch that is both protected from being accidentally tripped by other robots, and is accessible enough to be turned on and off quickly and easily. Team Whyachi makes a great master power switch for combat robots; it's a bit pricey, so builders may opt to modify a boat switch to be turned by a screwdriver.



Power switch.

BotTip: The most successful builders keep plenty of spare parts handy. That way, when a motor burns out or a wheel gets torn apart by a saw blade, it's a quick replacement in the pits, instead of the end of the line.

PARTS LIST

ARMOR & CHASSIS: Metal for the armor and chassis is most easily purchased locally, however, it can be purchased from numerous websites including [www.metalsdepot.com/products/hrsteel2.phtml?page=sheet&LimAcc=\\$LimAcc](http://www.metalsdepot.com/products/hrsteel2.phtml?page=sheet&LimAcc=$LimAcc).

SPEED CONTROLLERS: The IFI Victor 883 speed controller can be purchased at www.robotmarketplace.com/products/IFI-V883.html. NOTE: Remember to purchase the PWN booster cable AND the 24V fan (they're options in the drop-down menu).

MOTORS: The Magnum 775 Planetary Gearmotor can be purchased at www.robotmarketplace.com/products/RP-MAGNUM775.html.

WHEELS & HUBS: A selection of Colson wheels can be found at www.robotmarketplace.com/products/colson_wheels.html Hubs can be found at www.robotmarketplace.com/products/hubs.html Note: The Magnum 775 Planetary Gearmotor has a .50" shaft, meaning you'll need hubs with a .50" bore.

BATTERIES: The 18V 3.0 Ah NiCad BattlePack can be purchased at www.robotmarketplace.com/products/BPK-3000-18.html.

RADIO CONTROL TRANSMITTER & RECEIVER: The Spektrum DX6i six-channel 2.4 GHz Radio System Mode 2 can be purchased, along with the BR6000 Bot Receiver at www.robotmarketplace.com/products/O-SPM6600.html. The battery needed for the receiver can be purchased at www.robotmarketplace.com/products/O-SPM9520.html.

FUSE: A 60 amp fuse, along with a fuse holder, can be purchased at a local electronics store, or online at www.radioshack.com/product/index.jsp?productId=2102777&filterName=Category and www.radioshack.com/product/index.jsp?productId=2062255&filterName=Category, respectively.

LED INDICATOR LIGHT & RESISTOR: An LED indicator light and appropriate resistor can be purchased at a local electronics store, or online at www.radioshack.com/product/index.jsp?productId=2062549&filterName=Category and www.radioshack.com/product/index.jsp?productId=2062324, respectively. NOTE: When purchasing your LED and resistor, ask your vendor to double-check their compatibility, both with the 18V battery we'll be using and each other. Otherwise, you may blow out your LED.

MASTER POWER SWITCH: You can purchase a master power switch to your liking at your local electronics store, or find a nice selection at www.robotmarketplace.com/products/power_switches.html.

WIRES: 12-gauge wire can be purchased locally, or online at www.robotmarketplace.com/products/wire_and_accessories.html.

WIRES

While they may seem somewhat innocuous, it is important to give special attention to the wiring of our ComBot. We need to ensure we use a low enough gauge that the wires can stand up to the amp-flow of the batteries, but a high enough gauge that the wires are manageable. It's also important to make sure that the bot is wired in a way that there is a bit of slack so that during heavy impacts, the wires don't break loose. For our build, we'll use 12 gauge wire which can be picked up at any local electronics store.



Wire.

CHASSIS DESIGN

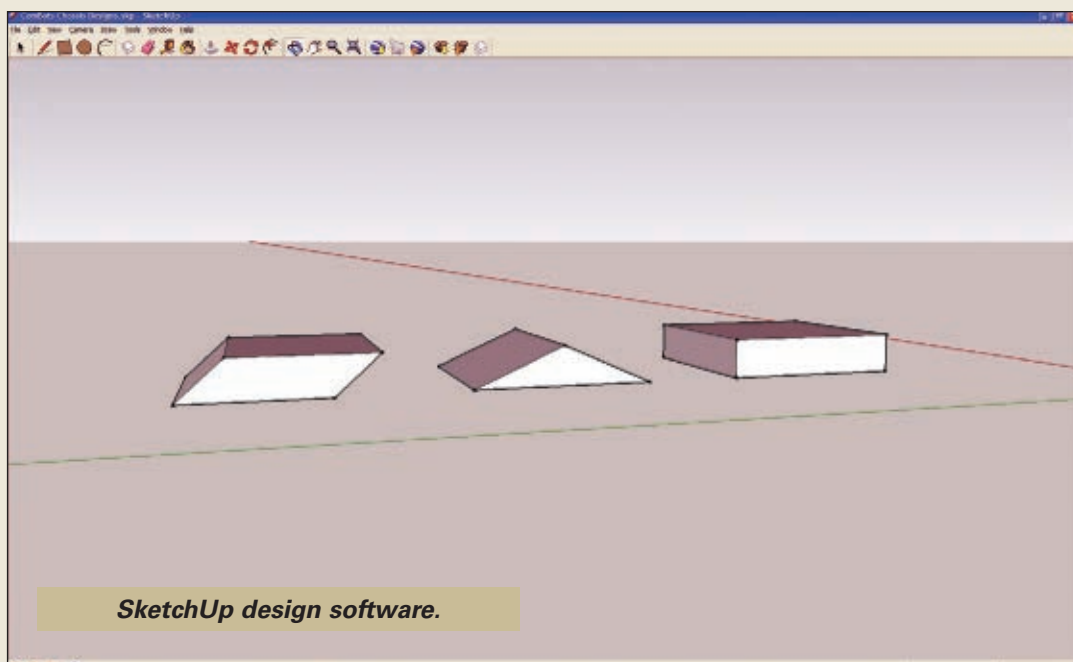
Before buying parts for a ComBot, it's important to have a design in mind. While you may not have the latest and greatest 3D design software, there's very little that you can't do with just a pen and paper. Sketch out the shape of the robot first. Is it a wedge? Where does the weapon come out of the chassis? Where do the motors go? How will the motors connect to the weapon and drive wheels? Always remember, KISS (Keep It Simple, Stupid!). The most successful ComBots are often the simplest, and it's very important to keep your first design clean and easy to build. For the sake of simplicity, let's keep this first design to a quadrilateral footprint.

In designing the ComBot, keep in mind that arcs and triangles are the strongest and second strongest shapes, respectively. That means that the design of the bot should incorporate those structures, especially at points of heavy impact. Again — to keep things simple — it's advisable to avoid arcs in the design unless you have easy access to machinery capable of producing such shapes.

The simple reality of ComBots is that the competing robots get beaten up during every match, and good robot designs should be easy compartmentalized so that instead of replacing an entire robot every time it takes a hit, only the portion that is damaged will need replacing. For this reason, it's extremely helpful to keep the chassis and armor as separate beings. In other words, we could spend eight hours milling out the perfect unibody chassis, but after it takes its first hit, it'll get bent out of shape, meaning another eight hours of milling. Instead, plan to

BotTip: Google SketchUp is a great, free tool to create a quick and dirty chassis design with.

BotTip: If your local metal works doesn't have the metal you need for your chassis, they're sure to know someone who does- so don't hesitate to ask!



bolt an outer armor to a frame-like chassis with a buffer of rubber between them to help absorb the shock. This way, it's easy to remove the armor after each match and either bang it back into shape or replace it. Keep in mind that the frame itself should be extremely sturdy so it doesn't bend out of shape.

Another thing to remember is the simplicity of component interaction. In other words, how hard is it to remove a battery? A speed controller? A motor? While we always want sturdy connections, it's a good idea to design the ComBot in a way that makes component removal quick and easy. Most bots attach all components to a common "base plate" — one big plate that also serves as armor for the underside of the ComBot. It's a great way to save time between matches since it eliminates extra steps when working with components.

Once you have a general chassis design sketched out, you'll want to do a bit of research to determine the exact dimensions of the components for the bot. Then, create a new sketch with everything to scale to ensure that it will all fit. Keep in mind that most 30 pound ComBots have a footprint somewhere in the neighborhood of 12" x 12 x 4" to 24" x 24 x 12" — if the bot is outside of that spectrum, it may be worth considering a redesign.

After completing the chassis design, it's time to purchase the required parts and components. Generally, the easiest way to get the metal needed for the chassis and armor will be via your nearest metal workshop. Google "metal works near [Your City]" and give them a

call. Before calling, though, have the exact dimensions in mind for the sheet metal cuts you'll need. In other words, if your armor is a 12 inch cube made of 1/16th inch steel, you'll need six 12" x 12" x 1/16" pieces of steel to build it.

THAT'S A WRAP!

In this first installment, we talked about the basics of building a 30 pound featherweight ComBot, and walked through the designing process. Next time, we'll cover how to put it all together, and include some tips and tricks of the trade. **SV**

RESOURCES

COMBOTS: ComBots is a combat robotics event for all ages held multiple times annually. The biggest ComBots event takes place at the International RoboGames competition. For more information on ComBots, visit <http://combots.net/>.

ROBOGAMES: RoboGames (formerly ROBOlympics) is an annual competition in the San Francisco Bay Area featuring robots from around the world competing in over 70 events. For more info on RoboGames, see <http://robogames.net/>.

SUICIDEBOTS: <http://SuicideBots.com> is a popular robotics blog, and one of the best sources of information regarding ComBots, RoboGames, and other awesome robot stuff.

GOOGLE SKETCHUP: SketchUp is a 3D modeling software from Google, and can be downloaded for free via <http://sketchup.google.com/>.

JUDGE DAVE'S GUIDE TO WINNING: Judge Dave's guide to winning is a short article by Dave Calkins — the founder of ComBots — on winning robotic combat competitions. The article can be found at www.robotics-society.org/jds-rules.shtml.

BotTip: In designing your robot, it's helpful to remember this baseline weight distribution:

- 30% (nine pounds) of total weight to the drive system.
- 30% (nine pounds) of total weight to the weapons.
- 25% (7-1/2 pounds) of total weight to the armor.
- 15% (4-1/2 pounds) of total weight to the batteries and electronics.

Swarm Robots and Sensor Virtualization

By Mike Keesling

What started out as a blurb about Zhu Zhu pets™ in *SERVO Magazine* has turned into a disturbingly large pile of Zhu Zhu pets. Some boxed, hopeful of a friendly home, others hacked apart, pelts laid aside like carcasses in some hideous abattoir. Fear not for them though, they are destined for greatness.

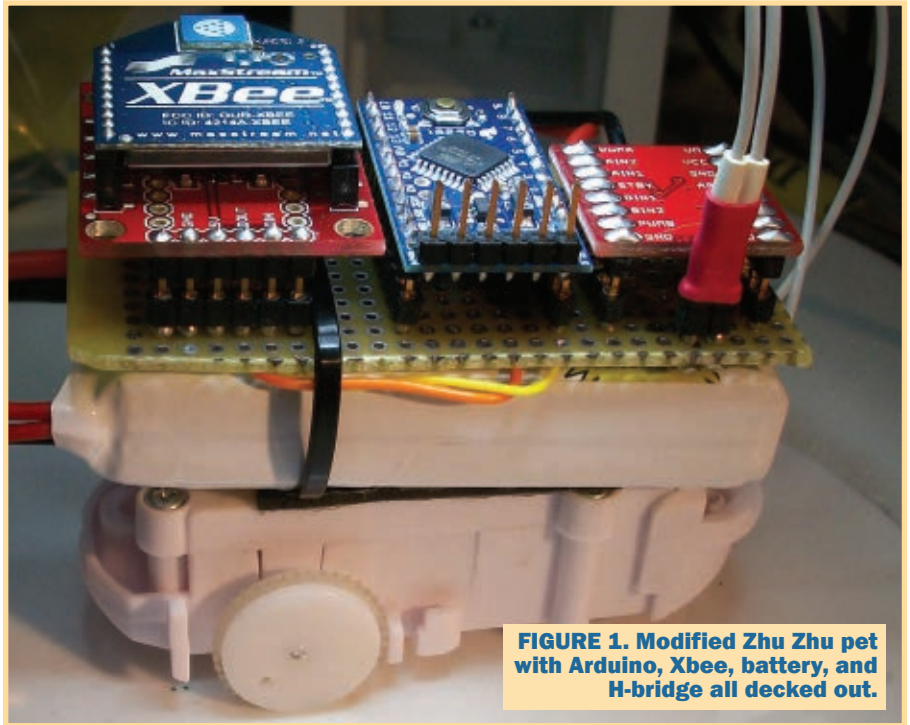


FIGURE 1. Modified Zhu Zhu pet with Arduino, Xbee, battery, and H-bridge all decked out.

It started with one labeled “Mr. Squiggles.” I bought him, hoping to find a simple differential drive platform that could be easily hacked. For \$10, I thought why not. I figured I would get two nice wheels at the very least. The sad truth is that Mr. Squiggles did not deliver my expected treasures. There was but a single motor, plus the wheels are deliberately cast off kilter to make the little artificial hamster wobble as he scuttles about.

So, how does a mock gerbil survive in a modern world with only one motor betwixt its two wobbly wheels? The same way cheap toys have been doing it for years: a turn on reverse mechanism. There is a clever little kick-stand which induces rotation when the motor runs in reverse. Certainly clever for a toy, and I suppose that a clever roboticist might cut their losses and move on, but I saw this as a challenge.

So, now — seven Zuh Zhu pets later — I have a \$70 challenge that I won’t be backing away from any time soon.

Anatomy of a Gerbil

The anatomy of this pesky plastic pest is at first a challenge, but there is a lot to be salvaged, right? Well, yes and no. The well-sighted and budget conscious among you can certainly make a lot out of all the artificial gerbil guts. There is a discreet H-bridge on the circuit board, some nice bump switches, and a speaker. I won’t be throwing these tiny treasures away, but I won’t be using them right now either. I am more interested in reliability and repeatability, and when hacking toy hamsters, you can’t count on either. I am also not terribly interested in cutting traces and injecting signals on a board that is small, optimized, and may be different if they source to different vendors. I have a lot of other things to accomplish first if this fleet is to grow in numbers (as hamsters are known to do).

So, new brains are most certainly a must-have, and a new motor drive is of great utility too. Throw in wireless communications and a microcontroller, and you have a lot of potential. I shopped around and noticed that the fine folks at SparkFun (www.sparkfun.com) had all that I needed, so I put together a big shopping list. They have a

lot of nice stuff, and some of it is amazingly inexpensive.

Parts List

- XBee Explorer regulated
- XBee Explorer USB (only one for main PC)
- XBee 1 mW Chip Antenna
- Motor Driver 1A Dual TB6612FNG
- Flex Sensor 4.5"
- Arduino Pro Mini 328 — 3.3V/8 MHz

In our hamster's inner workings, we have an Arduino orchestrating communications between the main host processor — in this case, a PC — and the H-bridge via the XBee. The Arduino runs a simple bit of code that acts to control the PWM signal to the H-bridge to give very simple acceleration and deceleration profiles. It also acts as a watchdog timer, shutting the motors down if no communications are received. Eventually, we will also install a battery monitor so that we do not destroy our precious Lithium polymer batteries. The XBee modules are communicating directly to the host, and no mesh networks are necessary.

The big ticket items here are the XBee wireless units to be sure, and there are less expensive ways to do things, but wireless is a place where I have been trained not to skimp on price. The next big ticket item is the Arduino. Honestly, for what we have planned, an Arduino is way overkill, but the Arduino has such a massive code base there is not much reason to replace it. In fact, the XBee does almost everything I need. This is where sensor virtualization comes in, as you will see later. For the most part, the Arduino will handle the motor drive PWM and maybe some blinking lights.

Additional mechanical and electrical concerns cropped up early on. The little kick-stand that causes the pets to lift a wheel off the ground and rotate when they back up is an area where we may experience difficulties. To counter any possibilities of our overall mass causing issues, we'll steer to Lithium polymer batteries. I opted for some nice packs from Common Sense (www.commonsense.com). They have high quality packs in a large variety of power and discharge

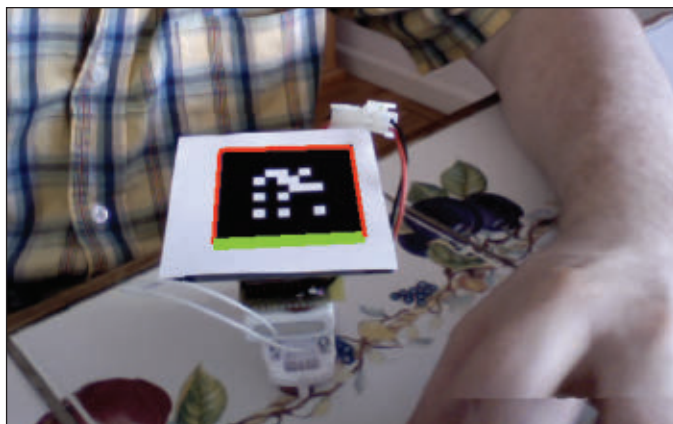


FIGURE 2. AR marker being recognized and overlaid with graphics.

ratings. These batteries may just save us from a total mechanical redesign.

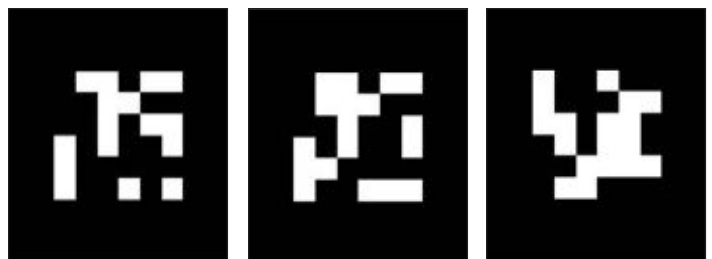
Making Sense of Things

Now that I'm well on my way to building an army, this mercenary of mischief is tired of being a lone soldier on a mission of madness. I need a general to help in the marshalling of my gerbil troops. For this foray to the edge of sanity, I look to my co-worker Scott Nichols for help with ideas and programming. I will leave the writing and building for myself.

Let's talk sensors. We want our gerbils to be well outfitted. Rotation sensors on the wheels to track position, maybe a magnetometer to sense absolute orientation. Range sensors to determine distance to other obstacles. Perhaps even cameras to better understand the immediate surroundings, or at the very least, some primitive color and light sensors. The problem is this all gets very expensive, complicated, battery hungry, and heavy very fast. This is where sensor virtualization comes in. For example, if I were to add an accelerometer, compass, several distance sensors, and wheel encoders, I could easily add \$200 to the budget for each hamster.

In concept, it is fairly simple. A camera mounted over the hamster habitat senses the position and orientation of every hamster in the environment through the use of fiduciary marks affixed to the tops of the hamsters. These fiducial marks actually provide complete six degree-of-freedom pose information that could, in fact, be used to provide a complete multimedia augmented reality experience. For now, though, I think we'll concentrate on just detecting the pose of the targets.

Doing some research into augmented reality has lead to a lot of stalled development. It seems like there isn't as much open source development going on as I had hoped. Most everyone involved in it seems to have gone the way of developing for private use or for mobile and industrial applications. This should not be a problem for us, though, since our needs are really quite modest compared to the capabilities that exist in the private sector. I did find the book *Augmented Reality: A Practical Guide* by Stephen Cawood for \$23 on Amazon. It promises a download of their SDK for private use, but the SDK isn't for distribution. Another option is ARtoolkitplus at http://studierstube.icg.tu-graz.ac.at/handheld_ar/artoolkitplus.php.



Any marker can represent anything. Here are a few for representational purposes.

Another possibility may be a platform called NyARToolkit, available at <http://sourceforge.jp/projects/nyartoolkit/> which promises ARtoolkit functionality with Java slowness and flexibility. Fortunately, our modest needs mean we can simply use ARtoolkit and do our programming in C.

The software functionality in the ARToolkit is very sophisticated. It first normalizes and thresholds the image to convert it to a binary image and then looks for any square shapes. The pose of the shapes, their position, and orientation relative to the camera is calculated. The interior is then analyzed and the symbols inside are recognized. Once the pose and identity of each marker is calculated, a transform can be applied to convert "camera coordinates" into "screen coordinates" so that graphics can be overlaid if you like.

The Nature of the Swarm

We have heard the term "swarm robotics" bandied about for some years now, but not many of us are experts in this somewhat esoteric field of study. There seems to be a lot to understand, but in reality there are only a few simple underpinnings that define a swarm.

Firstly, a swarm is comprised of multiple simple agents. These agents must be able to interact with each other in some way. If members of a swarm aren't aware of each other either through direct communications like sensing the behavior of nearby members via methods like the "honeybee dance" or indirectly via pheromone trails or some awareness of the effect the other members are producing, then you really aren't very "swarmy." Solitary bees, though similar to communal bees in anatomy, are not part of a swarm.

Secondly, members of a swarm are elements of a larger system. If your swarm is behaving well, you gain the benefits of scalability, redundancy, and robustness. You should be able to add or remove members of the swarm and it should still function properly. The collective effort of all these elements becomes a self-organizing system and will exhibit emergent behavior. Termites, for example, build vast homes for their colonies, defend, and forage, though no single termite is aware of the state of the entire mound.

Lastly, each member of the swarm needs to have: inputs for positive and negative feedback; a random element to their behavior; and (to drive the point) be able to interact with the environment in some way. Each member in a hive of wasps performs their part — whether hunting or building. They get their jobs done based on sensing and reacting to their local environment.

The word of the day: Stigmergy. It's fun to say, and it is the epitome of swarminess. Coined by French Biologist Pierre-Paul Grassé and defined as "Stimulation of workers by the performance they have achieved." Search it up on the Internet and it should lead you to a lot of interesting concepts in swarm intelligence.



Figure 3. Four AR markers delineating an arena with a Zhu Zhu pet inside.

Bringing It All Together

So, what do a couple of crazy roboticists and a handful of mechanical hamsters do to show swarminess? Scott came up with a really neat idea that leverages the virtual nature of the universe we are building. We will make a 4' x 4' playground that virtually tilts about its center. Because each marker — when sensed — conveys its "pose" in three dimensional space, an arena can be arbitrary. We picked 4x4 for convenience. There will be elements of virtual terrain that are more or less difficult to traverse, and the main driving force of the hamsters will be to sense the tilt of their world and act to bring it level.

In order to promote emergent behavior, positive feedback and communication will be through the actual tilt of the world. The hamsters will sense (virtually) the tilt and through some simple rules, try to bring the world into balance. Negative feedback will be the improper tilt of the world, the difficulty of the terrain, and the presence of other hamsters. Randomness will, in part, be the mechanical variations of their construction and random time variations of different mechanical processes that will be built into the software.

Being roboticists, the notions about feedback and communication are pretty easy to grasp. We have been adding positive and negative feedback to control systems since the days of the vacuum tube. What sounds counterintuitive at first is randomness.

Randomness sounds bad. How can you control something that behaves randomly? The necessity is subtle. Imagine 100 hamsters on this tilting flatland. All of them can move at the same speed and react the same way to input. If you inject an error, they all will react in a very similar way giving the very real possibility that the sum of all their reactions will result in an overall overcorrection to the tilt. Compounding this is they will now react to the overcorrection in unison again. This can lead to harmful resonances which can destroy the system. Random responses mean that the time in which any agent reacts will be slightly different, and the sum of all the agents will be closer to white noise than a pure resonant tone. **SV**



Then and NOW

ROBOT MOBILITY

b y T o m C a r r o l l

Recently, a young kid in my neighborhood came up to me and excitedly told me about several robots that he'd seen on television that could snake along the ground like a mechanical serpent and one that could climb up the side of buildings like Spiderman. We got to talking about how robots move around. The single feature that distinguishes a robot from a computer or any other sort of electro-mechanical device is its ability to move – whether the movement is an appendage or a movable base. Most of us want our creations to move about our house, yard or in some location. To accomplish this, we have to select and build a method of mobility. For now, and I'll concentrate on land vehicles only.

Robot Legs

The earliest robots of fiction used legs for mobility and were humanoid in form; they were *anthropomorphic* or 'man-formed.' It did not take robot experimenters long to discover that the technology required to develop a real walking robot was just not available in the early 20th

century. People have tried to make successful walkers for years but most seem to end up with overly large feet or "C" shaped feet that spread across a footprint that's wide enough past the center of gravity to prevent the robot from toppling over.

Accelerometers, gyros, and powerful variable-speed motors were not available in those days. Today, there are very successful walkers such

as Hubo, Asimo, and Big Dog.

When I was a kid, I built a walking robot from an Erector Set, jukebox, and other junk parts. I carefully trimmed the length of the leg segments (there were no knee joints) so that the resonance of the leg lengths would match the speed of the fixed speed AC motor. I finally got the robot to walk without falling over, but only if I held onto the power cord on start-up and stopping. When it started or stopped walking, it would always keel over.

University labs certainly built better walkers in those days, but they still were not the dynamically balanced machines we have today. Take, for example, Marc Raibert's 'Big Dog' shown in **Figure 1**. Walking robots have become quite popular in recent years, but I'm going to concentrate on other types of "mobile" robots in this article.

Early Robots Used Wheels

The first true experimental mobile robots used a wheeled platform base. This type of system seemed to be the logical choice as virtually every land-based machine that moved across a floor or the ground used wheels (except for other methods



FIGURE 1. Big Dog from Boston Dynamics.

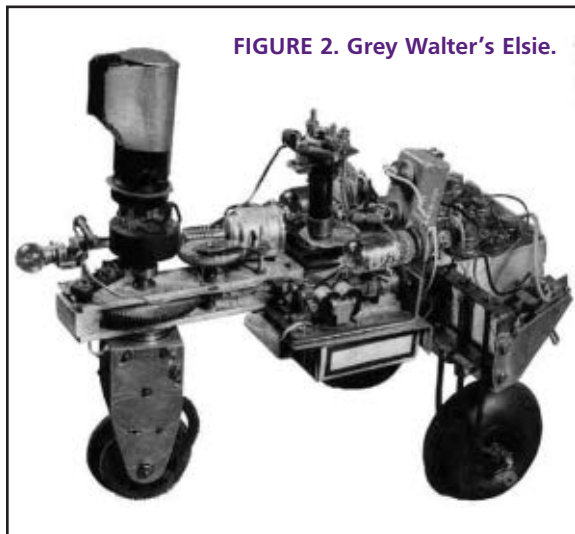


FIGURE 2. Grey Walter's Elsie.

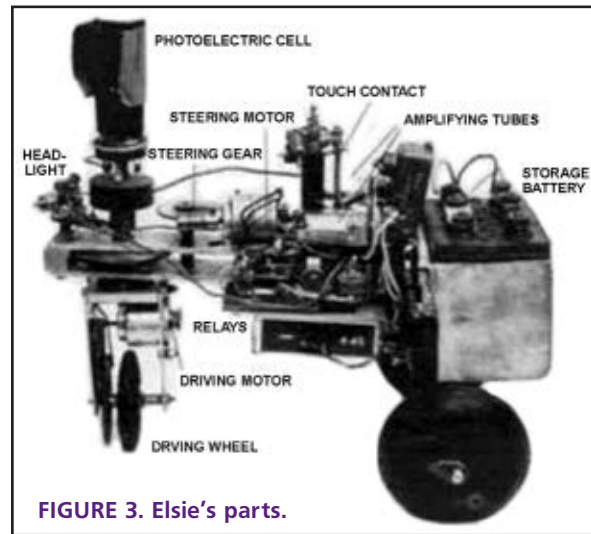


FIGURE 3. Elsie's parts.

like tank treads — but I'll cover that later.) Grey Walter's 'Elsie' robot in **Figure 2** from the late 1940s is an excellent example of an early experimental robot. You can see the worm drive motor used to steer the front wheel. **Figure 3** shows how the front wheel is driven by a motor mounted above the wheel. The steered single-wheel design proved to be quite functional since Elsie was used to demonstrate Pavlov's 'stimulus and response' theories.

This design was used long before the differential robot drives that many robot experimenters use today. Back in the mid '80s, many robot builders used a 6 VDC motorized wheel from kid's riding toys as a single driving motor.

Ackermann Steering

The Ackermann wheel and steering arrangement uses four wheels with either the front or back two wheels doing the steering. Some really great robot bases have been made with this setup and are especially popular in Robo-Magellan contests where the robots must travel over uneven outdoor terrain. This arrangement works very well if the robot is radio controlled and the operator is using one of the 'gun type' transmitters with a trigger to control speed and a steering wheel feature to control the car's direction.

Figure 4 shows Mark Curry's *Nomad* from the Robo-Magellan contest held last year at the Seattle Robotics Society (SRS) Robothon at Seattle Center. It was built on an R/C car chassis with Ackermann steering and was last year's winner.

Autonomous steering of this type of robot requires that several physical characteristics of the robot's base layout be identified. For example,

wheel diameter must be known to determine the distance traveled with one revolution of the wheel. Shaft encoders are usually placed on both drive wheels and an average number of encoder pulses determines the distance. True Ackermann steering — as in automobiles — actually has the two steered wheels turned at different angles. **Figure 5** shows how the turned inner wheel is angled more sharply due to the shorter turning radius of that wheel.

An encoder could also be used on the steering system to determine the turning angle, or just a single turning angle can be used. By utilizing

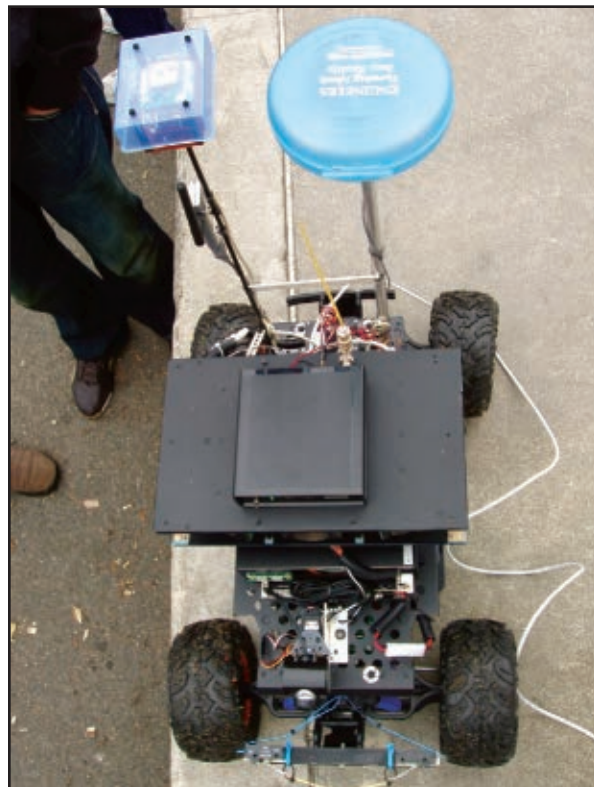
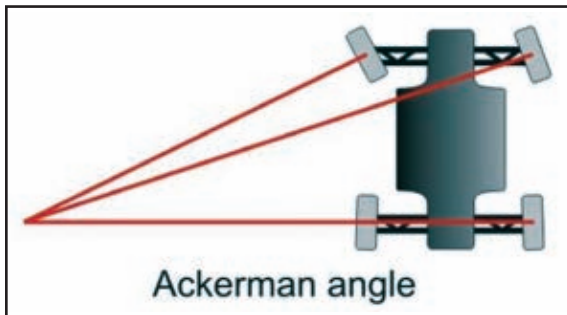


FIGURE 4. Top view of Nomad showing the Ackermann steering.

FIGURE 5.
Ackermann
steering
diagram (from
RC-setup.com).



a single right or left angle for the turns, the robot can be commanded to traverse in straight lines and then make the turns at intermediate points in the course. At each turning point, the steering motor drives the wheels to the hard point left or right; the drive wheels then turn a pre-programmed number of revolutions or until a compass tells the microcontroller that so many degrees have been turned. At this point, the wheels are turned back to forward and the robot continues to the next turning point. Both the gradual steering to make curved turns or the point-to-point style of steering can be very accurate, but will require a bit of programming expertise.

Differential Steering

Differential or 'tank style' steering is the most popular for experimental robots, especially with R/C robots as the steering can be done with a joystick on the R/C transmitter. Most builders of these types of robot rely on two side wheels with one caster or an omni-wheel in the front or back (or maybe one or more at each end). **Figure 6** from the SRS' website shows a simple LEGO robot with a single caster wheel.

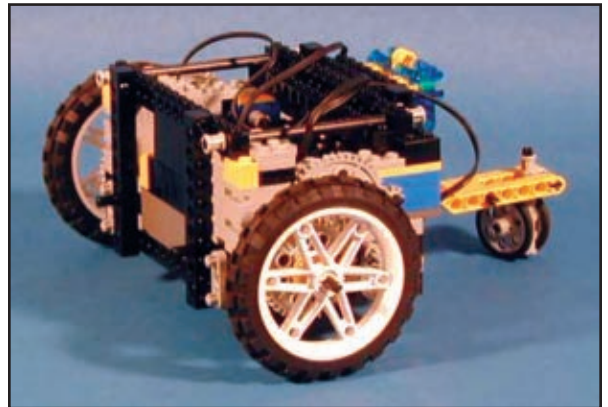


FIGURE 6. LEGO robot using caster wheel.

Steering is accomplished by driving the fixed side drive wheels at different speeds or directions. An example would be when the right wheel is slightly faster than the left, the robot makes a slight turn to the left. If the left wheel is not turning at all and the right wheel is moving, the robot makes a sharp turn to the left. In an extreme situation, if the right wheel is turning forward and the left wheel is turning backwards, the robot spins on its axis counter-clockwise.

Robot designers using both a fore and aft wheel must make sure that the center drive wheels are not high sided when traversing a concave surface. One or both casters can be spring-loaded enough to keep the robot from wobbling but not too much to prevent the drive wheels from touching the ground in a slight dip or hole.

Differential Skid Steering Uses More Power

Differential steering is also referred to as 'skid' steering. Mobile robots with differential steering have no problem with wheel friction if they use one or more swivel casters at one or both ends of the robot and only a *single* pair of drive wheels. If the robot designer tries to use two or more pairs of parallel drive wheels that only face in the fore-aft position, then skidding occurs when the robot is steered — just like when tank or bulldozer tracks skid on turning. An omni-wheel (as shown in **Figure 7**) has been used in many robot designs and is especially applicable when trying to minimize skid friction. Six-wheel differential drive robot platforms can use two standard drive wheels for the center wheel pair, and two other omni-wheel pairs for the fore and aft wheels. These wheels can also be powered, but can allow low friction side skidding in turns.

Skidding creates friction which equates to power loss. This type of skidding is not a problem



FIGURE 7.
Heavy-duty
omni-wheel from
Omniwheel.com.

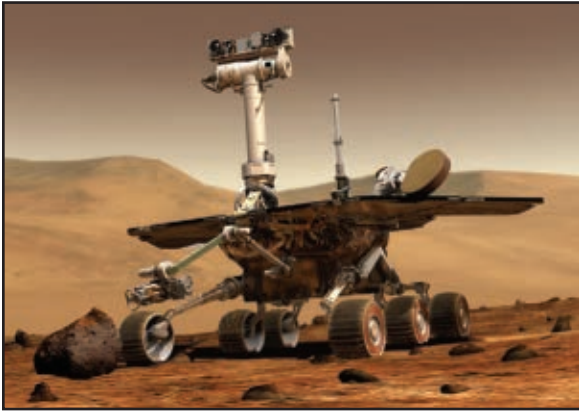


FIGURE 8. NASA's Spirit Rover depicted on Mars.

for combat robots that only need to operate for a few minutes at a time in a bout, but it sure eats up battery power for an experimenter's home robot. A four or six wheel robot with the wheels fixed may look really cool but the resulting skids consume a lot of excess power and can damage operating surfaces such as rugs and floor surfaces.

If at all possible, use a swivel caster or omni-wheel design. Radio control of a differential-configured robot is easy with a single joystick speed control such as the RDFR from Vantec.

Six Wheels on Mars

NASA engineers have used a combination of differential and Ackermann steering on several surface exploration robots for Mars including Spirit and Opportunity. Spirit (shown in **Figure 8**) has long outlasted its initial mission endurance of 90 days by operating on Mars for over six years. Spirit was launched in June of '03 and Opportunity launched a month later. The six wheel rovers utilize steerable wheels on the front and aft pairs — a necessity for robots operating from limited stored power derived from solar panels. Skid friction was not an option in the design process.

When a turn command is received from Earth, the front pair is turned in one direction and locked into position, then the aft wheels turn the same degree in the other direction and are locked to prevent steering drift during the turn. After the turn, the wheels are turned straight forward and locked into that position.

Spirit has been stuck in a sand trap for a year now, and there is little hope that it will be freed before the Martian winter sets in (or ever for that matter). One of the wheels died in 2006 and the other the end of last year. Opportunity is still mobile and both robots continue to transmit valuable data from Mars. The success of these two vehicles has given much credence to future robot



FIGURE 9. \$12K tank tread lawnmower from Craziestgadgets.com.

rover missions to other planets and moons in the solar system.

Tank Tracks for Robots

Differential steering cannot be adequately covered without mentioning tank treads or tracks. I've already discussed the skid friction problem that is virtually impossible to eliminate with tank-style tracks/treads, but this type of mobility has its place in robot design. Tank tracks have the greatest footprint on the ground and, therefore, have the greatest pulling power. Outside of Sumo competitions and hill climbing, pulling power is rarely the most desirable feature of a potential experimental robot design. One of the best features of a tracked robot is its ability to climb stairs. This is where track design shines. A string of articulated wheel sets with knobby treads can accomplish climbing in most cases, but tracks are best. Tracks are hard to fabricate by an amateur though, so most experimenters use tracks from toys. **Figure 9** shows a commercial radio-controlled 'robot' lawnmower priced at \$12,000 that uses tank treads. Priced at less than \$10, the Tamiya tank tread kit in **Figure 10** from **SparkFun.com** works great with



FIGURE 10. Tamiya tank treads from SparkFun.

Tamiya's dual gearmotor set, for example.

Are Two Wheels Better Than Four?

Self-balancing robots and various other technologies use gyros, accelerometers, and other sensors to keep a robot balanced. Back at the 2003 Robothon in Seattle, Larry Barello of the SRS gave an interesting paper and demonstration of his Gyrobot balancing robot shown in **Figure 11**. Gyrobot was a labor of love over several years. Barello enlarged the wheels and lengthened the chassis over time to bring the center of mass further away from the wheels; he used an AVR robot controller board as the processor.

How About One Wheel?

Figure 12 shows the Ballbot built by the Robotics Institute at Carnegie-Mellon University. A paper by B. Lauwers, G. A. Kantor, and R. L. Hollis of CMU describes this unique 'inverse mouse-ball drive' system that produced a human-sized robot that can interact with people. The power and balancing system are high in the structure (just like Barello's GyroBot) and it uses two drive motors that move the ball in two directions; this gives it movement capability in any direction so it can

Tom Carroll can be reached at
TWCarroll@aol.com.

respond to the balancing sensors. The ball allowed the robot to instantly move in any direction without having to turn first.

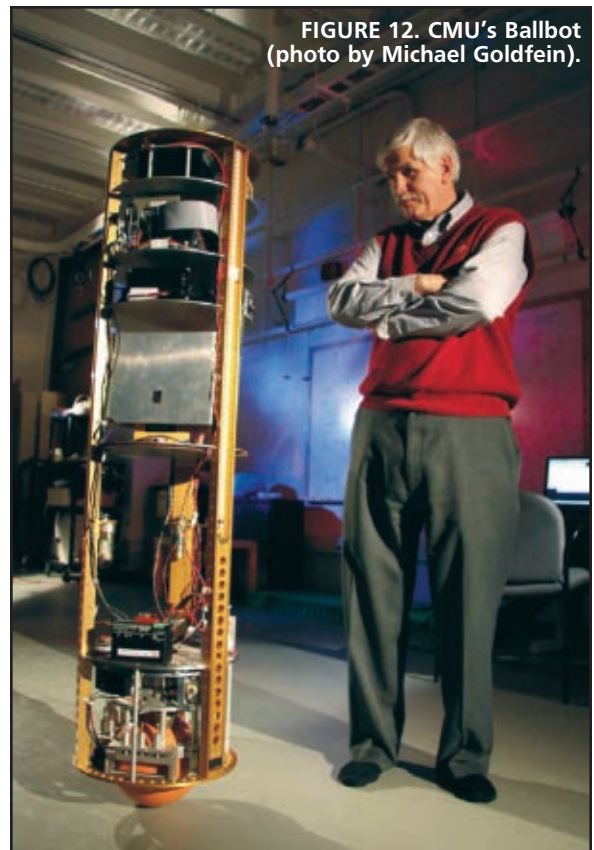
Final Thoughts

I've only touched on a few key mobility methods used by today's robot designers. Robot snakes that use serpentine propulsion techniques and robots with sucker feet that climb walls and glass windows for cleaning and inspection are a couple other methods to move your bot around. Pairs of rotating screws, hovercraft robots, claws to grasp the sides of trees, and dual-sided wheel arrangements to allow a robot to operate if it's flipped over are some other methods. The less expensive plastic omni-wheels have been used by robot experimenters for years and the advent of the Rovio brought forth a renewed interest. Are any of these methods that I've discussed the best? It really depends on your project and budget. If a propulsion method has been thought of by somebody, you can bet that a *SERVO* reader or robot experimenter somewhere has built a prototype of that design. **SV**

FIGURE 11. Larry Barello's Gyrobot.



FIGURE 12. CMU's Ballbot (photo by Michael Goldfein).



ROBO-LINKS

GPS MADE SIMPLE™

Linx Technologies.com

BiPOM Electronics, Inc. Your One Source for μ Controller Systems
ARM, AVR, PIC, 8051, 680X, STAMP and others

μ Controller Boards
Peripherals
Development Tools
Emulators
Programmers
Robotics

GadgetPC ARM9 USB
Computer with Linux

Sensors
Dev. Training Kits
Instruments
Displays

WebCat+
Embedded Web Server

www.bipom.com

RobotShop.com

superbrightleds.com

Component LEDs - LED Bulbs - LED Products

Super Bright LEDs Inc. St. Louis, Missouri - USA superbrightleds.com

THE ORIGINAL SINCE 1994

PCB-POOL®

Beta LAYOUT

- Low Cost PCB prototypes
- Free laser SMT stencil with all Proto orders

WWW.PCB-POOL.COM

IMAGES Scientific Instruments

SCIENCE, ROBOTICS & ELECTRONICS

Microcontrollers
Servo Control & Motors
Artificial Vision
Speech Recognition

www.imagesco.com

Tele: (718) 966-3694 Fax: (718) 966-3695

Pololu Robotics & Electronics

3095 E. Patrick Ln. #12, Las Vegas, NV 89120

www.POLOLU.com 1-877-7-POLOLU

Motor, servo, and robot controllers
Sensors

Robots, motors, wheels, and more!

CAT CAN

www.catcan.com.tw

SMART CONTROL BOARD
SMART CONTROL BOARD
SMART CONTROL BOARD
SMART CONTROL BOARD

Beginner's Guide To Embedded C Programming

SERVO MAGAZINE

Book & PIC Kit2 Combo

www.servomagazine.com

\$29.95 MaxSonar-EZ1 (MSRP)

High Performance
Ultrasonic Range Finder

- serial, analog voltage & pulse width outputs
- lowest power - 2mA
- narrow beam
- very easy to use!

www.maxbotix.com

The most comprehensive resource for news and links

THE ROBOT REPORT

TRACKING THE BUSINESS OF ROBOTICS

www.TheRobotReport.com

For the finest in robots, parts, and services, go to www.servomagazine.com and click on **Robo-Links**.

ALL ELECTRONICS CORPORATION

Electronic Parts & Supplies Since 1967

NUBOTICS™

www.nubotics.com

Distributed by Acroname.com

LINEAR SERVOS

Firgelli

www.firgelli.com

ADVERTISER INDEX

All Electronics Corp.	27, 81	Images Co.	81	Robotis	2
AP Circuits	26	Linx Technologies	81	RobotShop, Inc.	81, 82
BaneBots	41	Lynxmotion, Inc.	83	Solarbotics/HVW	21
Basic Micro	26	Maxbotix	81	Superbrightleds.com	81
BiPOM Electronics	81	Nu-botics	27, 81	The Robot Report	81
Critical Velocity	27	Parallax, Inc.	41	Vantec	23
Digilent	Back Cover	PCB Pool	23, 81	WeirdStuff Warehouse	27
Firgelli	27, 81	Pololu Robotics & Electronics ..	22, 81	Yost Engineering	3
Galaxy Far East Corp.	81	Robot Power	36		
Gears Educational Systems	7	RoboGames	37		



Robots, lots of robots!

The World's Leading Source
for Domestic and Professional Robot Technology



www.robotshop.com



The Lynxmotion Servo Erector Set Imagine it... Build it... Control it!

Featured Robot

**Biped BRAT does
Mech Warfare!**

Introducing the Hunchback!
The world's first completely
functional biped Mech Warrior!

They said a BRAT
based Mech couldn't
be done. That it was too much
payload. That it would require
expensive digital servos. Well, they
were wrong! This robot project was made
from a BRAT with HS-645MG servos and
standard off the shelf components!

We have created tutorials on how to
make a Mech Warrior including code.
Now anyone can make a Biped Mech!

Youtube videos
User: Robots7



Biped Nick



Biped Pete



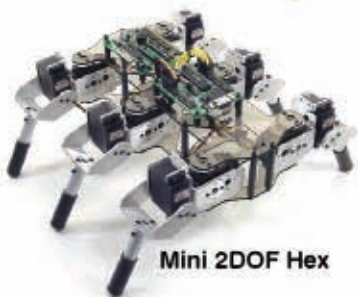
Biped Scout



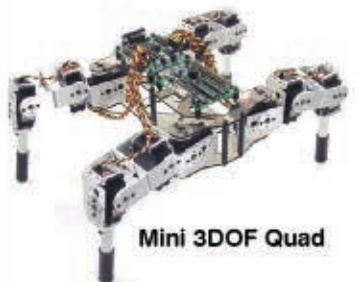
Biped 209



Walking Stick



Mini 2DOF Hex



Mini 3DOF Quad

**With our popular Servo Erector Set you can easily
build and control the robot of your dreams!**

Our interchangeable aluminum brackets, hubs,
and tubing make the ultimate in precision
mechanical assemblies possible. The images here
are just a sample of what can be built. The Bot
Board II and SSC-32 provide powerful control
options. Our Visual Sequencer program provides
powerful PC, Basic Atom, or BS2 based control.



Bot Board II - \$24.95
Carrier for Atom / Pro, BS2, etc.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
Buffered Speaker.
Sony PS2 game controller port.
3 Push button / LED interface.



SSC-32 - \$39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
No better SSC value anywhere!

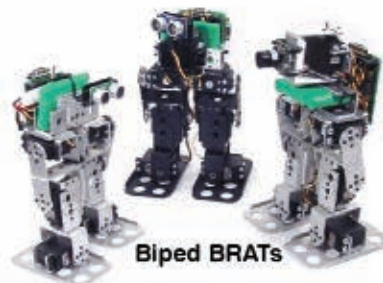
We also carry motors, wheels, hubs, batteries, chargers,
servos, sensors, RC radios, pillow blocks, hardware, etc!



Visit our huge website to see our complete line
of Aluminum and Lexan based robot kits,
electronics, and mechanical components.



CH3-R Hexapod



Biped BRATs



Phoenix



Images represent a fraction of what can be made! www.lynxmotion.com The SES now has 157 unique components!

Face it. You've never been like the others.



We know how you feel.

CEREBOTTM
32MX4

\$54

Enter the value code "servo2010a" on our website for this special introductory price

A powerful PIC32[®] based microcontroller board, ideal for robotic experimentation in C, C++, and Assembler.

- Microchip[®] PIC32MX460F512L
- 80 MHz 32-bit MIPS processor
- 512K Flash, 32K RAM
- Eight R/C servo connectors, two SPI ports, two I²C ports, two UARTs
- Nine 12-pin Pmod[™] Connectors
- Five 16-bit timers with 5 input capture and 5 PWM outputs
- 16 channel, 10-bit, 500kps A/D converter
- On-board USB Program / Debug Interface
- USB OTG Host / Device Capable
- USB Powered



Pmod[™]
Peripheral Modules

\$9.99 - \$29.99

Expand your designs

Pmods are a wide range of small, inexpensive & versatile I/O interface boards.



PmodJSTK
2-axis joystick
w/ 3 pushbuttons



PmodHBS
2A H-bridge
w/ feedback inputs



PmodENC
Rotary encoder module
w/ integral pushbutton



PmodMIC
Electret mic w/
compressor, & 12-bit A/D



PmodCLS
2-line character LCD
w/ serial interface



Digilent, Inc. is a leader in the design & manufacture of FPGA and microcontroller technologies. Our products can be found in over 100 countries and over 2000 universities worldwide. Visit our website for a wide assortment of these boards, as well as peripherals, reference designs, sample projects, tutorials, textbooks, and more.

www.digilentinc.com